NUMERICAL EXAMPLES IN THE INVESTIGATION OF A PARTICULAR MATRIX IN EIGENVECTOR THEORY

ALBERT N. YOST

NUMERICAL EXAMPLES IN THE INVESTIGATION OF A PARTICULAR MATRIX IN EIGENVECTOR THEORY

* * *   * * *

Albert N. Yost

NUMERICAL EXAMPLES IN THE INVESTIGATION OF A PARTICULAR MATRIX IN EIGENVECTOR THEORY

by

Albert N. Yost
Lieutenant, United States Navy

Submitted in partial fulfillment of
the requirements for the degree of

MASTER OF SCIENCE
IN
AERONAUTICAL ENGINEERING

United States Naval Postgraduate School

Monterey, California

1 9 6 5

NUMERICAL EXAMPLES IN THE INVESTIGATION OF A PARTICULAR MATRIX IN EIGENVECTOR THEORY

by

Albert N. Yost

This work is accepted as fulfilling

the thesis requirements for the degree of

MASTER OF SCIENCE

IN

AERONAUTICAL ENGINEERING

from the

United States Naval Postgraduate School

ABSTRACT

For every orthonormal matrix X there exists a skew-symmetric matrix N such that $X = (I-N)(I+N)^{-1}$, provided that $(I+N)^{-1}$ exists. A matrix K, similar to N, can be defined for biorthonormal matrices U and V such that $U = (I-K)(I+K)^{-1}$, provided that $(I+K)^{-1}$ exists. Numerical methods are presented for examination of the properties of K. The particular property anticipated for K, that it exhibit n(n-1) basic parameters inherent in biorthonormal matrices, is not apparent in the numerical examples derived.

TABLE OF CONTENTS

1. INTRODUCTION

If A is a real symmetric matrix of order n, it possesses n real orthogonal eigenvectors x_i which may be normalized, as described in Appendix 1. The square modal matrix X' of these eigenvectors,

$$X' = (x_1 \; x_2 \; . \; . \; . \; x_n) \tag{1}$$

is orthonormal, that is:

$$\tilde{X}'X' = I = X'\tilde{X}' \tag{2}$$

For the purpose of this analysis the eigenvalues of A are assumed to be discrete and the eigenvectors unique.

The n^2 elements of such an orthonormal matrix are subject to the conditions of equation (2), of which only $n(n+1)/2$ are independent, leaving $n(n-1)/2$ independent parameters by which the orthonormal matrix can be expressed. Consequently, as has been shown by Heading [3] for example, X' can be written in general as

$$X' = (I-N)(I+N)^{-1}, \tag{3}$$

provided that $(I+N)^{-1}$ exists, where N is skew-symmetric and contains the required $n(n-1)/2$ parameters. Hence any real skew-symmetric matrix defines an orthonormal matrix, provided that $(I+N)$ is non-singular. Equation (3) may be solved for $N(X')$:

$$N = (I+X')^{-1}(I-X') \tag{4}$$

The proof of equation (3) hinges upon the fact that $I+N$ and $I-N$ commute, i.e.

$$\begin{aligned}\tilde{X}'X' &= (I-N)^{-1}(I+N)(I-N)(I+N)^{-1}\\ &= (I-N)^{-1}(I-N)(I+N)(I+N)^{-1}\\ &= I\end{aligned}$$

since $(I+N)(I-N) = I-N^2 = (I-N)(I+N)$; furthermore N must be skew-symmetric in order that $I-\tilde{N} = I+N$ as used above.

The discussion of this property of orthonormality in the context of

the eigenvalue problem is of particular significance in the reversed problem suggested by Bell [4]. The solution of his problem for A symmetric is made possible by the existence of N, as in equation (3), containing just the right count of parameters required for the solution. As pointed out in [4], the $n(n+1)/2$ independent elements of a symmetric A should yield the same total number of independent parameters in the eigenvectors and eigenvalues of A. This exact count does appear in the n eigenvalues plus the $n(n-1)/2$ elements of N.

This description of the property N(X'), or N(A) for A symmetric as the property is restated in the preceding sentence, is background for the work to be reported herein. In [4] Bell has shown that an attempt to extend his analysis to a class of unsymmetric matrix A, possessing a real eigenvector matrix U', would involve the property K(U') analogous to N(X'), where the matrix K should express $n(n-1)$ independent parameters inherent in U', or twice as many as are inherent in X'. The development of this concept is presented in the next section, where the multiplicity possible for K is pointed out, as is the need for a definition, if possible, of that canonical form, K_c, which displays the required property.

Numerical methods are set up in the present investigation for deriving K(A), with A unsymmetric, and for exploring its properties in association with those of the modal matrix U', and of the modal matrix V' which is adjoint to U'. These methods are programmed for arbitrary matrices of order up to n = 25.

2. PROBLEM STATEMENT

If A is real but unsymmetric, then its eigenvectors u_i are not orthogonal; but the adjoint set of vectors v_i, belonging to A, satisfies

a biorthogonality with the u_i, as described in Appendix 2. In fact the square matrices of these adjoint vectors,

$$
\begin{aligned}
U' &= (u_1\ u_2\ .\ .\ .\ u_n) \\
V' &= (v_1\ v_2\ .\ .\ .\ v_n)
\end{aligned}
\tag{5}
$$

satisfy the conditions of biorthonormality:

$$\tilde{V}'U' = I = U'\tilde{V}', \tag{6}$$

if the u_i and v_i are mutually normalized.

It should be noted here that, as before, the eigenvalues of A are assumed to be discrete and the eigenvectors unique; A is also restricted to that class for which the eigenvectors are real.

The $2n^2$ elements of such real adjoint matrices in (5) are subject to the n^2 conditions of equation (6), all of which are independent, indicating that both U' and V' can be expressed with no more than n^2 basic parameters. As shown in [4], U' and V' can in fact be written as

$$
\begin{aligned}
U' &= (I-K)(I+K)^{-1} \\
V' &= (I+K)(I-K)^{-1}
\end{aligned}
\tag{7}
$$

provided $(I+K)^{-1}$ exists, where K is an arbitrary real matrix. Hence, by equations (7) any real matrix defines a pair of real adjoint or biorthonormal matrices U' and V' under the provisions stated. Equations (7) may be solved for K:

$$
\begin{aligned}
K &= (I+U')^{-1}(I-U') \\
\tilde{K} &= (V'+I)^{-1}(V'-I)
\end{aligned}
\tag{8}
$$

It is readily verified that equations (7) and (8), for A symmetric,

are identical with equations (3) and (4), since for A symmetric:

$$\begin{aligned} U' &= V' = X' \\ K &= N = -\tilde{K} \end{aligned} \tag{9}$$

The proof of equations (7) follows from the commutability of (I+K) and (I-K), as for equation (3). The consistency of equations (8) is demonstrated for the general case in [4].

The diagonal elements of I in equation (6) are the normalized inner product, or dot product, of the i^{th} columns, or vectors, of U' and V'. Unlike the case for the vectors x_i in equation (2), this is not a unique specification for the dot product of u_i into itself, or of v_i into itself, i.e., for the "magnitudes" of u_i and v_i. For each such pair of adjoint vectors u_i and v_i (belonging to the same eigenvalue of A) there is a free condition to be chosen arbitrarily. There are n such pairs and hence n such arbitrary conditions. This suggests that the property K(U') can involve not as many as n^2 basic parameters, but rather as few as n(n-1). An additional argument given in [4], on the basis of the eigenvalue problem itself, is similar to the one given in the Introduction for symmetric A, namely: an unsymmetric matrix A has n^2 independent elements and should yield just as many total basic parameters in its eigenvalues, which are n in number, and in its eigenvectors, which must therefore have only n^2-n.

As shown in Appendix 2, there are an infinite number of pairs of biorthonormal matrices, satisfying equation (6), belonging to a given matrix A. This multiplicity corresponds to the fact that the eigenvectors can be multiplied by any scalar and still satisfy the eigenproblem statement, coupled with the free condition on the dot product mentioned in the preceding paragraph. In view of equations (8), there must be a corresponding multiplicity in K(U') for any A, or in other words in K(A).

If K can be found to involve as few as $n(n-1)$ independent parameters, the skew-symmetric form which K takes with A symmetric suggests that in the more general case these $n(n-1)$ parameters would be off-diagonal elements, and that the main diagonal would remain empty. This form of K, if it should occur only for one particular pair of U' and V' out of the infinite multiplicity possible, might be designated its canonical form, K_c, corresponding to the canonical pair, U_c' and V_c'. The conditions on U' and V' for the canonical form, if it exists, have not yielded easily to analysis for $n \geq 4$, but might become apparent from a study of numerical examples.

The problem which was stated for this investigation may now be outlined as follows:

1. Set up the computer solution for K from a given unsymmetric A, using equations (8).

   This has involved the use of known techniques for diagonalizing A; Laguerre's method [5] was adapted. It was also necessary to develop subroutines for numerical solutions of the adjoint vectors U' V', when the eigenvalues by Laguerre's method were known. Care must be taken that the given A possess real eigenvectors. The solutions for U' and V' are verified for biorthonormality in equation (6).

2. Set up the computer solution for U' and V' from arbitrary K, using equations (7).

   This is provided as a final subroutine to Step 1, for an overall check of the solution; it is also provided as a separate program for the investigation of U' and V' from arbitrary K. The latter program also includes solutions for $\tilde{V}'V'$, $\tilde{U}'U'$, $V'\tilde{V}'$, $U'\tilde{U}'$, and $\tilde{V}'U'$ (see Appendix 6) to provide for possible interpretations of the numerical relationships among the adjoint vectors themselves.

The objectives of Step 1 were first to provide an overall check of the methods developed in [4] for A unsymmetric, and second to exhibit numerically some examples of the property $K(A)$, with particular regard for evidence of its basic parameters. The objectives of Step 2 were to explore whether evidence of the conditions on these parameters might be exhibited

readily in the numerical characteristics of U' and V', for example in or among the "magnitudes" of the eigenvectors, as displayed in the elements of the diagonals of $\widetilde{U}'U'$ and $\widetilde{V}'V'$. The computer programs for these investigations are written in Fortran 63 and can accommodate matrix orders of 25 or less.

The matrix properties which are being sought out in this investigation are known to have hypergeometric interpretations for n greater than three [4]. For n less than three they represent the familiar interpretations of Euclidean three-space. Attempts to extrapolate the latter into higher order hyperspace can be misleading. On the other hand, it was considered desirable to keep n small at the outset in order to limit numbers of numerical data to be examined. For this reason the numerical examples reported herein are for third and fourth order matrices only.

A third computer program was formulated in order to further investigate the characteristics of the matrix K. In this case the K matrix, however, was derived from an orthonormal matrix after the latter had been premultiplied by a diagonal matrix. The goal here was to observe the effects of premultiplication in the changes on the matrix K. Appendix 3 describes in detail the background for this program.

3. COMPUTER PROGRAMS

Two Fortran 63 computer programs were formulated in order to satisfy the requirements set forth in the Problem Statement above, Steps 1 and 2. All programs were arranged to operate with matrices of order $n \leq 25$.

Appendix 5 describes the computer program named REVEIG, designed for Step 1, which basically is a general eigenvalue program for any real square matrix A where, as described in Appendix 2,

$$AU' = U'D$$

$$\tilde{A}V' = V'D$$

and $$\tilde{V}'U' = I$$

The derived matrices U' and V' are then applied in equations (8) to derive K and $\tilde{K}$, from U' and V' respectively.

The available EIG3 subroutine used to derive the eigenvalues for REVEIG can be used to find real and imaginary roots of a matrix and store them for further use [5]. A basic assumption for REVEIG was that all matrices investigated would be required to have real roots only. Subroutine EIG3, however, did not derive the eigenvectors along with the eigenvalues. A program for the eigenvectors was derived as an additional step, as described in Appendix 5. It should be noted here that due to the arbitrariness of the solution used for the eigenvectors, a particular restricted class of multiplicity is built into the program, to the exclusion of other classes which exist, and which might bear further investigation. Some typical test results from REVEIG are contained in Appendix 8.

Appendix 6 describes the computer program called REVEIG1, designed for Step 2, which was based on equations (7) in the Introduction. Inputs were made to REVEIG1 from the K and $\tilde{K}$ matrices derived in REVEIG, in order to provide a check of REVEIG1 by recomputing the U' and V' matrices from which K was derived. Thereafter, other arbitrary matrices could be used as K inputs to REVEIG1. As a part of this program the products $\tilde{V}'V'$, $V'\tilde{V}'$, $\tilde{U}'U'$, $U'\tilde{U}'$, and $\tilde{V}'U'$ are also derived. Some typical results are shown in Appendix 8.

Appendix 7 describes the computer program called REVEIG2. This program was formulated late in the investigation in order to further explore the effects of matrix algebra on an orthonormal matrix. As described in

Appendix 7, the orthonormal matrix U' derived in REVEIG was premultiplied by a diagonal matrix G of significance in a particular case. The result was used as an input to derive the matrix K using equation (8) in the Introduction. The ultimate goal was to investigate the effects of premultiplying an orthonormal matrix in this special case, and to determine what characteristics, if any, exist in the resulting K matrix. The significance of this case is discussed in Appendix 3.

The three computer programs are designed for an input format of (4E20.10). The outputs are all printed out in a (7E17.10) format. In order that the computer results in Appendix 8 be more easily understood, the following relationships are defined:

Computer	Theory
UP	U'
VP	V'
KT	$\tilde{K}$
UPT	$\tilde{U}'$
VPT	$\tilde{V}'$

Comment cards are included in the program decks for all the programs, that basically describe the individual steps involved. The program decks can be acquired from the Department of Aeronautics of the U. S. Naval Postgraduate School.

4. RESULTS AND CONCLUSIONS

A few of the several numerical examples run from the programs described are included in Appendix 8, showing typical results. It was a primary objective of this investigation to provide an overall numerical check of the methods developed in [4] for A unsymmetric, namely to derive a property K(A) as described in Section 2. In this objective the investigation is successful, and the consistency of equations (7) and of equations (8) is verified.

A second objective was to exhibit typical examples of the matrix K with the view that typical patterns might be evident in its properties. In particular the so-called canonical form dependent upon n(n-1) basic parameters was of interest. This form was not observed, at least not in the form with empty main diagonal, and was not otherwise identified. In this connection it should be noted that a single restricted class of multiplicity has been built into REVEIG, as discussed in Section 3.

Inputs were made into REVEIG1 consisting of arbitrarily assumed "canonical" forms, K_c, possessing zero main diagonal. It was anticipated that the final printout of this program, consisting of the several products listed in equation (2-8) of Appendix 2, might show evidence of associated canonical properties in U' and V'. These properties were not observed.

Finally a calculation with REVEIG2 was made to explore the form of K which results from a special case of unsymmetric matrix, as described in Appendix 3. No significance was found in the resulting example.

BIBLIOGRAPHY

1. Crandall, S. H. Engineering Analysis. McGraw-Hill, 1956.

2. Lanczos, C. Applied Analysis. Prentice-Hall, Inc. 1956.

3. Heading, J. Matrix Theory for Physicists. Longmans, Green and Co., Ltd. 1958.

4. North American Aviation, Inc., Los Angeles Division. A Feasibility Study on the Generation or Prediction of Structural Influence Coefficients to Meet Simple Aeroelastic Criteria, by R. W. Bell. January, 1964. Report No. NA-63-1183.

5. Beresford Parlet, Mathematics of Computation, American Mathematical Society, Laguerre's Method Applied to the Matrix Eigenvalue Problems, Vol. 18, No. 87, July 1964: 464-485.

NOMENCLATURE

Most terms within this thesis are defined when used; however, the following list of the general terms used is supplied for ready reference.

A = real square matrix

a_{ij} = elements of A

A^* = $B^{-\frac{1}{2}}AB^{-\frac{1}{2}}$ = a derived matrix in Appendix 3

B = diagonal matrix of non-vanishing, real diagonal elements

D = diagonal matrix of eigenvalues assumed to be distinct

E = diagonal matrix

F = diagonal matrix

G = diagonal matrix

I = identity matrix

K = matrix property belonging to A unsymmetric

K_c = hypothetical canonical form of K

n = integer

N = skew-symmetric matrix

u_i = eigenvector (column matrix)

U = modal square matrix of eigenvectors u_i

U_c = hypothetical canonical form of U

v_i = eigenvector (column matrix)

V = modal square matrix of eigenvectors v_i

V_c = hypothetical canonical form of V

W = diagonal matrix

x_i = eigenvector (column matrix)

X = orthogonal modal matrix of eigenvectors x_i

X^* = pseudomodal matrix (Appendix 3)

λ_i = real, discrete eigenvalues of A

$\widetilde{(\ )}$ = transpose of ( )

$(\ )'$ = normalized form of matrix ( )

APPENDICES

APPENDIX 1

SIMPLE EIGENVALUE PROBLEM WITH A SYMMETRIC

The simplest possible eigenvalue problem is defined [1,2] by

$$Ax_i = \lambda_i x_i \tag{1-1}$$

where (see Appendix 4 for definition of terms):

A = n x n symmetric matrix

λ_i = eigenvalue for which a solution x_i exists; there are n solutions assumed to be discrete

x_i = column matrix (eigenvector)

The n solutions may be combined into a single statement

$$AX = XD \tag{1-2}$$

where

X = modal matrix of eigenvectors

D = spectral matrix (diagonal) of eigenvalues

It is to be noted here that post-multiplication of the modal matrix by a diagonal matrix does not alter the eigenproblem (1-2).

After premultiplying equation (1-2) by $\tilde{X}$, and also after transposing equation (1-2) and postmultiplying by X:

$$\begin{aligned} \tilde{X}AX &= \tilde{X}XD \\ &= D\tilde{X}X \end{aligned}$$

Putting $\tilde{X}X = W$ and subtracting: $WD - DW = 0$, i.e. $W_{ij} = 0$, $i \neq j$, since the λ_i are assumed discrete. Therefore

$$\tilde{X}X = W \text{ (diagonal)}$$

i.e., X is orthogonal. Letting $X' = XW^{-\frac{1}{2}}$, i.e., after normalization:

$$\tilde{X}'X' = I$$

Note also that $$X'\tilde{X}' = I$$

In this case X' is said to be orthonormal.

APPENDIX 2

EIGENVALUE PROBLEM WITH A UNSYMMETRIC: GENERAL CASE

If A is real but unsymmetric its eigenvectors are not orthogonal (self-adjoint) as in Appendix 1. However A and $\tilde{A}$ have the same eigenvalues (assumed to be discrete) but different eigenvectors, v_i and u_i respectively [2]. The n solutions for each may be combined as follows:

$$AU = UD \tag{2-1}$$

$$\tilde{A}V = VD \tag{2-2}$$

where (see Appendix 4 for definition of terms):

A = n x n real unsymmetric matrix

U = Modal n x n matrix of eigenvectors u_i associated with A

V = Modal n x n matrix of eigenvectors v_i associated with A

D = Spectral (diagonal) matrix of eigenvalues

Again as in Appendix 1, after premultiplying (2-1) by $\tilde{V}$, and also after transposing (2-2) and post-multiplying by U:

$$\tilde{V}AU = \tilde{V}UD \tag{2-3}$$

and

$$\tilde{V}AU = D\tilde{V}U \tag{2-4}$$

Putting $\tilde{V}U = W$ and subtracting:

$$0 = WD - DW$$

i.e. $W_{ij} = 0$, $i \neq j$, since the λ_i are assumed discrete. Therefore

$$\tilde{V}U = W \text{ (diagonal)} \tag{2-5}$$

or U and V are said to be biorthogonal.

Since the basic eigenproblem

$$Au_i = \lambda_i u_i$$

is unchanged by multiplication with a scalar, it follows that any scalar multiple of u_i is a solution. Hence if U and V are solutions to (2-1)

and (2-2), respectively, so also are UE and VF, where E and F are arbitrary real diagonal matrices (but note that only post-multiplication by E and F is permissible). It is clear that by putting

$$E = F = W^{-\frac{1}{2}} \tag{2-6}$$

then

$$U' = UE = UW^{-\frac{1}{2}}$$
$$V' = VF = VW^{-\frac{1}{2}}$$

will have been mutually normalized, i.e.

$$\tilde{V}'U' = I \tag{2-7}$$

In this case U' and V' are said to be biorthonormal; they are also said to be adjoint.

It will be noted that if (2-7) is satisfied, then by transposition and commutation it is also true that

$$U'\tilde{V}' = \tilde{U}'V' = V'\tilde{U}' = I + \tilde{V}'U' \tag{2-8}$$

It is also clear that (2-6) is not a unique formula for mutual normalization of U and V. The values

$$E = W^{-2},\ F = W$$

would have done as well, or any of an infinite variety of combinations, i.e., there is an infinite multiplicity in the pairs of matrices which satisfy equations (2-8) and which are adjoint eigenvectors of A.

APPENDIX 3

A PARTICULAR EIGENVALUE PROBLEM

A special case of unsymmetric matrix is derived from the following eigenproblem. Let

$$AU' = BU'D \tag{3-1}$$

where (see Appendix 4 for definition of terms)

A = real symmetric matrix

B = real diagonal matrix of non-vanishing diagonal elements

U' = modal matrix (real)

D = spectral matrix (real)

This can be converted to the form of equation (2-1), Appendix 2, since B^{-1} exists:

$$CU' = U'D \tag{3-2}$$

where $C = B^{-1}A$ is not symmetric. Therefore, as in Appendix 2, U' is not orthonormal, but the adjoint V' exists such that $\tilde{V}'U' = I$, and the property K(C) can be defined as in the text, equations (8).

On the other hand since A and B in equation (3-1) are both symmetric, it is possible by factoring and multiplying through by $B^{-\frac{1}{2}}$ to rearrange equation (3-1) as follows [4]:

$$\underbrace{B^{-\frac{1}{2}}AB^{-\frac{1}{2}}}_{A^*}\ \underbrace{B^{\frac{1}{2}}U}_{X'^*} = \underbrace{B^{\frac{1}{2}}U}_{X'^*}\ D \tag{3-3}$$

where

A^* = symmetric = $B^{-\frac{1}{2}}AB^{-\frac{1}{2}}$

X'^* = orthonormal pseudomode = $B^{\frac{1}{2}}U'$ (3-4)

D = spectral matrix, unchanged

As in Appendix 1, since A^* is symmetric then X'^* must be orthonormal:

$$\widetilde{X'}^*X'^* = I$$

It follows that the property N(A*) exists for X'*, as in the text, equation (4). But by equation (3-4) this property N(A*) can also be associated with U. Since the property K(C) is also associated with U, then the question is suggested of how N(A*) and K(C) may be related. This question can be amplified as follows. Since B, or $B^{\frac{1}{2}}$, or $B^{-\frac{1}{2}}$, is arbitrary (diagonal), then equation (3-4) may as well be written

$$U' = GX'^* \tag{3-5}$$

where

$$\begin{aligned} G &= B^{-\frac{1}{2}} \qquad (3\text{-}6) \\ &= \text{arbitrary diagonal matrix} \end{aligned}$$

The conversion back to the context of equation (3-1), for some given G, may be carried out at any time. The general question can now be stated, namely, given X'*, an orthonormal matrix with skew-symmetric property N(X'*), pre-multiplied as in (3-5) by any G: does the property K belonging to U' = GX'* have the canonical form sought in the text, for example does the main diagonal of K remain empty?

APPENDIX 4

DEFINITION OF TERMS

Vector..........A quantity in an n^{th} order system represented by a column matrix:

$$\begin{pmatrix} x^1 \\ x^2 \\ \cdot \\ \cdot \\ \cdot \\ x^n \end{pmatrix}$$

Modal matrix ....A composition of all possible eigenvectors into one square matrix. The columns of eigenvectors are formed side by side.

Transpose ....The transpose of a matrix A, written $\tilde{A}$, has for its i^{th} row the i^{th} column of A and for its j^{th} column the j^{th} row of A, for all rows i and columns j. $\tilde{A}$ is said to be obtained from A by interchanging rows and columns.

Symmetric........If $\tilde{A} = A$, A is said to be symmetric. A symmetric matrix must be square and its elements satisfy $a_{ij} = a_{ji}$, for all i and j. The matrix is symmetric with respect to its main diagonal.

Skew-symmetric...If the square matrix A is such that $\tilde{A} = -A$, it is said to be skew-symmetric. In a skew-symmetric matrix, $a_{ij} = -a_{ji}$. In particular, the elements a_{ii} of the main diagonal must be zero.

Spectral matrix..A diagonal matrix with eigenvalues (λ_i) as its main diagonal.

Orthogonal.......If the product of a real matrix X with its transpose is a diagonal matrix, W, then X is said to be orthogonal: $\tilde{X}X = W$. Each vector (column) of X is orthogonal to all other vectors (columns) of X.

Orthonormal......A non-singular matrix X is said to be orthonormal if $X^{-1} = \tilde{X}$; thus an orthogonal matrix X which has been normalized is said to be orthonormal: $\tilde{X}X = I$.

Biorthogonal.....U and V are biorthogonal if $\tilde{V}U = W$ (diagonal).

Biorthonormal....If $\tilde{V}U = I$, then V and U are biorthonormal.

APPENDIX 5

PROGRAM REVEIG

The first program was based on the analysis of Appendix 2, to solve for U and V from a given A:

$$AU = UD \tag{5-1}$$

$$\tilde{A}V = VD \tag{5-2}$$

where A = Square matrix up to order 25x25.

U = Modal matrix of eigenvectors with respect to A.

V = Modal matrix of eigenvectors with respect to A.

D = Diagonal matrix of characteristic eigenvalues.

The subroutine used for the eigenvalues is the standard EIG3 subroutine, which solves for the roots of an n^{th} degree polynomial. The eigenvectors are not supplied by this standard subroutine. It was necessary to program for them a solution of the homogeneous equations $A - \lambda I = 0$ for each eigenvalue; this was facilited by arbitrarily setting one coordinate of each eigenvector equal to unity. The program then tests each diagonal element of $W = \tilde{V}U$ for algebraic sign. If W contains negative elements, the sign of this element and of the respective column of U is reversed, before U is printed out, to permit real square roots of W in accordance with equation (2-6) of Appendix 2, in the next step. The change of sign is permissible because only direction of the eigenvector is affected thereby.

Biorthonormality was then effected by determining a new U and V, called UP and VP in the print-out, as follows:

$$(UP) = UE \tag{5-3}$$

$$(VP) = VF \tag{5-4}$$

where E and F both equal $W^{-\frac{1}{2}}$. The product $\tilde{V}U$ was then printed out using

the new UP and VP to check for biorthonormality.

Following equations (8) of the text the K matrix and its transpose were then determined from U' and V', respectively, using the following relationships

$$K = (I+UP)^{-1}(I-UP) \tag{5-5}$$

$$\tilde{K} = (VP+I)^{-1}(VP-I) \tag{5-6}$$

These must, of course, be consistent derivations of K; it was partially the purpose of this investigation to verify this fact numerically.

The program was tested with both a 3x3 and a 4x4 symmetric A, in which case UP = VP as was expected, in accordance with equations (9) of the text.

```
OQP,,YOST,S/1S/2S/E/6=51,5,5000.
TN,L,E.
      PROGRAM REVEIG
     0DIMENSION A(25,25),U(25,25),AT(25,25),V(25,25),RTR(25),RTI(25),AUX
     1(25,25),W(25,25),E(25),F(25),UP(25,25),VP(25,25),VPT(25,25),FK(25,
     225),FKT(25,25),AUX1(25,25),RHS(25)
      EQUIVALENCE (UP,W)
GIVEN MATRIX A, FIND THE U MATRIX(ASSOCIATED EIGENVECTORS) AND D(DIAGONAL MAT
IX OF ASSOCIATED EIGENVALUES) USING EIG3 SUBROUTINE WHICH BASICALLY USES NTH
EGREE POLYNOMIALS TO SOLVE THE PROBLEM.  EIG3 ALSO RETAINS THE REAL ROOTS AND
IMAGINARY ROOTS FOR LATER USE IF REQUIRED.
    1 READ 999,N
  999 FORMAT (I5)
      IF (N) 2,2,3
    2 STOP
    3 DO10 I=1,N
      READ 998,(A(I,J),J=1,N)
   10 PRINT998,(A(I,J),J=1,N)
  998 FORMAT (4E20.10)
      DO 15 I=1,N
      DO 15 J=1,N
   15 AUX(I,J)=A(I,J)
      CALL EIG3 (AUX,N,N,RTR,RTI,25)
      EIG3 WILL PRINT INTERMEDIATE RESULTS AND RTR,RTI
      DO 11 I=1,N
      IF(ABSF(RTI(I))-1.E-5)111,11,12
   12 PRINT 997
  997 FORMAT(29H0EIG3 SAYS EIGVAL IS COMPLEX.)
      STOP
  111 RTI(I)=0.
   11 CONTINUE
REARRANGE EIGENVALUES SO THA, SMALLEST APPEARS FIRST
FORM U MATRIX SINCE EIG3 SOLVES FOR EIGENVALUES ONLY.
ASSUME ONE COORDINATE OF EACH SET OF EIGENVECTORS EQUALS ONE.  USE EQUATION A
LAMBDA *I=0
      CALL RERTR(N,RTR)
      DO 13 I=1,N
      DO 14 J=1,N
      DO 14 K=1,N
      AUX(J,K)=A(J,K)
      IF(J-K)14,130,14
  130 AUX(J,K)=AUX(J,K)-RTR(I)
   14 CONTINUE
      NM1=N-1
      DO 140 K=1,NM1
  140 RHS(K)=-AUX(K,N)
      CALL MATALG(AUX,RHS,NM1,1,0,DET,25)
   16 DO 18 J=1,NM1
   18 U(J,I)=RHS(J)
      U(N,I)=1.
   13 CONTINUE
      PRINT 995
  995 FORMAT (9H0U MATRIX/)
      DO 150 I=1,N
  150 PRINT 991,I,(U(I,J),J=1,N)
      CALL MATRA (N,N,A,AT,25,25)
```

```
      DO 19 I=1,N
      DO 19 J=1,N
   19 AUX(I,J)=AT(I,J)
C     DO SAME AS BEFORE USING AT VICE A, VVICE U
C ALSO TRANSPOSE A AND DETERMINE THE V MATRIX OF EIGENVECTORS AND THE SAME DIA
C NAL MATRIX OF EIGENVALUES USING SAME EIG3 SUBROUTINE.
      CALL EIG3(AUX,N,N,RTR,RTI,25)
      DO 20 I=1,N
      IF(ABSF(RTI(I))-1.E-5)120,20,21
   21 PRINT 994
  994 FORMAT(29H0EIG3 SAYS EIGVAL IS COMPLEX.)
      STOP
  120 RTI(I)=0.
   20 CONTINUE
C REARRANGE EIGENVALUES SO THAT SMALLEST APPEARS FIRST
      CALL RERTR(N,RTR)
      DO 22 I=1,N
      DO 23 J=1,N
      DO 23 K=1,N
      AUX(J,K)=AT(J,K)
      IF(J-K)23,230,23
  230 AUX(J,K)=AUX(J,K)-RTR(I)
   23 CONTINUE
      DO 231 K=1,NM1
  231 RHS(K)=-AUX(K,N)
      CALL MATALG(AUX,RHS,NM1,1,0,DET,25)
   25 DO 27 J=1,NM1
   27 V(J,I)=RHS(J)
      V(N,I)=1.
   22 CONTINUE
      PRINT 992
  992 FORMAT(9H0V MATRIX/)
      DO 220 I=1,N
  220 PRINT 991,I,(V(I,J),J=1,N)
C TEST V TRANSPOSE TIMES U TO SEE IF OFF-DIAGONALS ARE ZERO WHICH ENSURES ORTH
C ONALITY.  CALL RESULT W.
      CALL MATRA(N,N,V,AUX,25,25)
   56 CALL MAMUL(N,N,N,AUX,U,W,25,25,25)
      PRINT 990
  990 FORMAT (9H0W MATRIX/)
      DO 28 I=1,N
   28 PRINT 991,I,(W(I,J),J=1,N)
  991 FORMAT(I5/(1X7E17.10))
C   ASSUME W IS A DIAGONAL MATRIX
C ARRANGE W SO THAT THE DIAGONALS ARE ALWAYS POSITIVE BY CHANGING SIGNS OF RES
C CTIVE COLUMNS OF THE U MATRIX.  PRINT U ANX W MATRICES AGAIN.
      LLL=1
      DO 50 L=1,N
      IF(W(L,L)) 51,50,50
   51 DO 52 LL=1,N
   52 U(LL,L)=-U(LL,L)
      LLL=2
   50 CONTINUE
      GO TO (54,53),LLL
   53 PRINT 995
      DO 55 I=1,N
```

```
 55 PRINT 991,I,(U(I,J),J=1,N)
    GO TO 56
C TAKE MINUS SQUARE ROOT OF W AND STORE IN E AND F.
 54 DO 30 I=1,N
    WW=W(I,I)
    EF=SIGNF(SQRTF(ABSF(WW)),WW)
    EF=1./EF
    E(I)=EF
 30 F(I)=EF
C POSTMULTIPLY U BY E AND V BY F AND PRINT U*E AS UP AND V*F AS VP.
    CALL MAMUL1(N,N,U,E,UP,25,25)
    PRINT 980
980 FORMAT (15HOU-PRIME MATRIX/)
    DO 31 I=1,N
 31 PRINT 991,I,(UP(I,J),J=1,N)
    CALL MAMUL1(N,N,V,F,VP,25,25)
    PRINT 981
981 FORMAT (15HOV-PRIME MATRIX/)
    DO 32 I=1,N
 32 PRINT 991,I,(VP(I,J),J=1,N)
C TRANSPOSE VP
    CALL MATRA(N,N,VP,VPT,25,25)
C MULTIPLY VPT TIMES UP TO CHECK FOR ORTHONORMALITY BY SEEING IF RESULT IS IDEN
CITY MATRIX.
    CALL MAMUL(N,N,N,VPT,UP,AUX,25,25,25)
    PRINT 982
982 FORMAT(32HOV-PRIME-TRANSPOSE TIMES U-PRIME/)
C COMPUTE K AND KT MATRIX WITH GIVEN UP AND VP USING THE FOLLOWING EQUATIONS, K
C (I+UP) INVERSE TIMES (I-UP)  AND  KT= (VP+I) INVERSE TIMES (VP-I)
    DO 33 I=1,N
 33 PRINT 991,I,(AUX(I,J),J=1,N)
    DO 40 I=1,N
    DO 40 J=1,N
    IF (I-J) 413,414,413
414 AT(I,J)=1.+UP(I,J)
    GO TO 40
413 AT(I,J)=+UP(I,J)
 40 CONTINUE
C  USE AT AS TEMPORARY STORAGE
    CALL MATALG(AT,AUX1,N,N,1,DET,25)
    DO 41 I=1,N
    DO 41 J=1,N
 41 AUX(I,J)= -UP(I,J)
    DO 42 I=1,N
 42 AUX(I,I) =AUX(I,I)+1.
    CALL MAMUL(N,N,N,AUX1,AUX,FK,25,25,25)
    PRINT 983
983 FORMAT (9HOK MATRIX/)
    DO 43 I=1,N
 43 PRINT 991,I,(FK(I,J),J=1,N)
    DO 44 I=1,N
    DO 44 J=1,N
    IF (I-J) 415,416,415
416 AT(I,J)=1.+VP(I,J)
    GO TO 44
415 AT(I,J)=VP(I,J)
```

```
 44 CONTINUE
USE AS TEMPORARY STORAGE
    CALL MATALG(AT,AUX1,N,N,1,DET,25)
    DO 45 I=1,N
    DO 45 J=1,N
 45 AUX(I,J)=VP(I,J)
    DO 46 I=1,N
 46 AUX(I,I)=AUX(I,I)-1.
    CALL MAMUL(N,N,N,AUX1,AUX,FKT,25,25,25)
    PRINT 984
984 FORMAT (10HOKT MATRIX/)
    DO 47 I=1,N
 47 PRINT 991,I,(FKT(I,J),J=1,N)
    GO TO 1
    END
    SUBROUTINE EIG3(A,N,M,RTR,RTI,NQ)
          474 10152 EIGENVALUES OF REAL MATRICES
051/1 EIGENVALUES OF NON-SYMMETRIC MATRICES
    DIMENSIONA(NQ,NQ),NC(100),RTR(M),RTI(M)
    CALL OVFSET                                                      ADDED
    TRACE=A(1,1)
    DO 10 I=2,N
 10 TRACE=TRACE+A(I,I)
    WRITE OUTPUT TAPE 6,4,TRACE
    CALL TRING(A,1.E-7,N,NC, NQ)
    TRACE=A(1,1)
    DO 11 I=2,N
 11 TRACE=TRACE+A(I,I)
    WRITE OUTPUT TAPE 6,4,TRACE
    NU=0
    NV=0
 13 IF (NV-N)14,12,14
 14 NV=NV+1
    NU=NV
 16 IF (NC(NV))15,17,15
 15 NV=NV+1
    GO TO 16
 17 IF (NV-NU)19,18,19
18   RTR(NU)=A(NU,NU)
     RTI(NU)=0.
     WRITE OUTPUT TAPE 6,5,RTR(NU),RTI(NU)
     GO TO 13
 19 IF (NV-NU-1)20,21,20
20   NP=XMINOF(M,NV)
     CALL LAGER(A,1.E-4,NP,NU,NV,RTR,RTI,NQ)
     GO TO 13
 21 RR=.5*(A(NU,NU)+A(NV,NV))
     E1=RR**2-A(NU,NU)*A(NV,NV)+A(NU,NV)*A(NV,NU)
     S=SQRTF(ABSF(E1))
     IF(E1)22,23,23
23   RTR(NU)=RR+S
     RTI(NU)=0.
     RTR(NV)=RR-S
     RTI(NV)=0.
25   WRITE OUTPUT TAPE 6,5,RTR(NU),RTI(NU),RTR(NV),RTI(NV)
     GO TO 13
```

```
 22    RTR(NU)=RR                                                       EIG300
       RTI(NU)=S                                                        EIG300
       RTR(NV)=RR                                                       EIG300
       RTI(NV)=-S                                                       EIG300
       GO TO 25                                                         EIG300
 12    X=0.                                                             EIG300
C      CALL FPOLD                                                       EIG300
       DO 24 J=1,M                                                      EIG300
 24    X=X+RTR(J)                                                       EIG300
       WRITE OUTPUT TAPE 6,6,X                                          EIG300
       RETURN                                                           EIG300
     4 FORMAT(1H048X,7HTRACE =E16.8)                                    EIG300
     5 FORMAT (11HOEIGENVALUE 12X,2E20.8)                               EIG300
 6     FORMAT(1H035X,20HSUM OF EIGENVALUES =E16.8)                      EIG300
       END                                                              EIG300
C051/2ALMOST TRIANGULAR (HESSENBERG) SUBROUTINE                          EIG300
       SUBROUTINE TRING(A,EPS,N,INT, NQ)
       DIMENSION A(NQ,NQ),    INT(NQ)
       WRITE OUTPUT TAPE 6,1                                            EIG300
       N1=N-1                                                           EIG300
       N2=N-2                                                           EIG300
       DO 21 J=1,N1                                                     EIG300
       S=ABSF(A(J,J+1))                                                 EIG300
       J1=J+1                                                           EIG300
       J2=J+2                                                           EIG300
       L=J1                                                             EIG300
       NJ1=N-J1                                                         EIG300
       IF (NJ1) 15,15,6                                                 EIG300
 6     DO 12 K=J2,N                                                     EIG300
       T=ABSF(A(J,K))                                                   EIG300
       IF(T-S)12,12,11                                                  EIG300
 11    L=K                                                              EIG300
       S=T                                                              EIG300
 12    CONTINUE                                                         EIG300
       IF(L-J1)13,15,13                                                 EIG300
 13    DO 131 K=1,N                                                     EIG300
       T=A(K,J+1)                                                       EIG300
       A(K,J+1)=A(K,L)                                                  EIG300
 131   A(K,L)=T                                                         EIG300
 14    DO 141 K=1,N                                                     EIG300
       T=A(J+1,K)                                                       EIG300
       A(J+1,K)=A(L,K)                                                  EIG300
 141   A(L,K)=T                                                         EIG300
 15    IF(S-EPS*MIN1F(ABSF(A(J,J)),ABSF(A(J+1,J+1))))16,16,17           EIG300
 16    L=0                                                              EIG300
       NJ1=0                                                            EIG300
       GO TO 181                                                        EIG300
 17    T=A(J,J+1)                                                       EIG300
       DO 18 K=J2,N                                                     EIG300
 18    A(J,K)=A(J,K)/T                                                  EIG300
 181   DO 20 I=1,N                                                      EIG300
       M=MINOF(J,I-2)                                                   EIG300
       U=0.                                                             EIG300
       IF (NJ1) 19,19,7                                                 EIG300
 7     DO 8 K=J2,N                                                      EIG300
 8     U=U+A(K,I)*A(J,K)                                                EIG300
```

```
9      IF (M) 20,20,9                                          EIG300
9      DO 10 K=1,M                                             EIG300
10     U=U-A(K,I)*A(J+1,K+1)                                   EIG300
20     A(J+1,I)=A(J+1,I)+U                                     EIG301
21     INT(J)=L                                                EIG301
22     INT(N)=0                                                EIG301
       RETURN                                                  EIG301
1      FORMAT(1H048X,22HALMOST TRIANGULAR FORM)                EIG301
       END                                                     EIG301
051/3LAGUERRE METHOD                                           EIG301
       SUBROUTINE LAGER(A,EPS,N1,NU,N,RTR,RTI,NDI)
       DIMENSION A(NDI,NDI),P(6,101),RTR(NDI),RTI(NDI),B(6)
       WRITE OUTPUT TAPE 6,1                                   EIG301
       ONCE=0.                                                 EIG301
       BL1=1.                                                  EIG301
       NUQ=NU-1                                                EIG301
       LLY=0                                                   EIG301
       DELOLD=1.                                               EIG301
       ROLD=1.                                                 EIG301
       EGSUM1=0.                                               EIG301
       EGSUM2=0.                                               EIG301
       CALL FPTEST(Z)                                          EIG301
       DO 10 L=2,6                                             EIG301
10     P(L,NU)=0.                                              EIG301
       NU1=NU+1                                                EIG301
       CUP=0.                                                  EIG301
       DO 11 J=NU1,N                                           EIG301
11     CUP=CUP+ABSF(A(J-1,J))                                  EIG301
       CUP=CUP/FLOATF(N-NU)                                    EIG301
       CAP=0.                                                  EIG301
       P(1,NU)=1.                                              EIG301
                                                               EIG301
       FIND TRACE OF H AND H SQUARED                           EIG301
                                                               EIG301
       SPUR1=A(NU,NU)                                          EIG301
       SPUR2=A(NU,NU)**2                                       EIG301
       DO 13 J=NU1,N                                           EIG301
       SPUR1=SPUR1+A(J,J)                                      EIG301
13     SPUR2=SPUR2+A(J,J)**2+2.*A(J-1,J)*A(J,J-1)              EIG301
                                                               EIG301
       INITIAL ITERATE FROM INFINITY                           EIG301
                                                               EIG301
14     S1R=EGSUM1-SPUR1                                        EIG301
       S2R=SPUR2-EGSUM2                                        EIG301
       F1=N-NUQ                                                EIG301
       IF(ABSF(S1R)+ABSF(S2R)-1.E-7*CAP)15,15,16               EIG301
15     XBAR=CUP                                                EIG301
       YBAR=0.                                                 EIG301
       GO TO 23                                                EIG301
16     F2=F1-1.                                                EIG301
       DR=F2*(F1*S2R-S1R**2)                                   EIG301
       ER=SQRTF(ABSF(DR))                                      EIG301
       IF(DR)17,18,18                                          EIG301
17     XBAR=-2.*S1R/F1                                         EIG301
       YBAR=2.*ER/F1                                           EIG301
       GO TO 23                                                EIG301
```

```
   18 YBAR=0.                                                          EIG301
      F2=SIGNF(1.,S1R)                                                 EIG301
      IF(S1R)22,20,22                                                  EIG301
   20 F2=0.                                                            EIG301
   22 XBAR=-(S1R+F2*ER)/F1                                             EIG301
C                                                                      EIG301
C     EVALUATE POLYNOMIAL AND DERIVATIVES                              EIG301
C                                                                      EIG301
   23 IF(ABSF(YBAR)-ABSF(XBAR)*1.E-6)24,25,25                          EIG301
   24 YBAR=0.                                                          EIG301
   25 M=6                                                              EIG301
      IF(YBAR)27,26,27                                                 EIG301
   26 M=3                                                              EIG301
 27   DO 34 K=NU,N                                                     EIG301
      T=-A(K,K+1)                                                      EIG301
      DO 34 L=1,M                                                      EIG301
      S=SIGNF(1.,3.5-FLOATF(L))                                        EIG301
      M1=L+3*XFIXF(S)                                                  EIG301
      R=-XBAR*P(L,K)+YBAR*S*P(M1,K)-FLOATF(XMODF(L-1,3))*P(L-1,K)      EIG301
      DO 28 J=NU,K                                                     EIG301
 28   R=R+P(L,J)*A(K,J)                                                EIG301
      CALL OVFTST (Z)                                                  ADDED
      IF(Z)29,32,29                                                    EIG301
 29   Z=0.                                                             EIG301
      P(1,NU)=1.E-10*P(1,NU)                                           EIG301
      IF(P(1,NU))30,30,27                                              EIG301
 30   F=FLOATF(K-NU)/FLOATF(N-NU+1)                                    EIG301
      WRITE OUTPUT TAPE 6,2,XBAR,YBAR,P(1,NU),F,ONCE                   EIG301
      XBAR=XBAR*F                                                      EIG301
      YBAR=YBAR*F                                                      EIG301
      P(1,NU)=1.0                                                      EIG301
      GO TO 27                                                         EIG301
 32   IF(N-K)33,33,34                                                  EIG301
 33   T=1.0                                                            EIG301
 34   P(L,K+1)=R/T                                                     EIG301
C                                                                      EIG301
C     SCALE DOWN                                                       EIG301
C                                                                      EIG301
      DO 39 K=1,6                                                      EIG301
   39 B(K)=0.                                                          EIG301
      DO 35 J=1,M                                                      EIG301
   35 B(J)=P(J,N+1)                                                    EIG301
      G1=ABSF(B(1))+ABSF(B(4))                                         EIG301
      G2=ABSF(B(2))+ABSF(B(5))                                         EIG301
      G3=ABSF(B(3))+ABSF(B(6))                                         EIG301
      WRITE OUTPUT TAPE 6,2,XBAR,YBAR,G1,G2,G3                         EIG301
 2    FORMAT(8H ITERATE20X,E15.8,5X,E15.8,8X,3E15.4)                   EIG301
      D=ABSF(B(1))                                                     EIG301
      DO 36 K=2,M                                                      EIG302
   36 D=MAX1F(D,ABSF(B(K)))                                            EIG302
      CALL SCALE(D,B,M)                                                EIG302
      IF (G1) 41,41,43                                                 EIG302
C                                                                      EIG302
C     REMOVE KNOWN ROOTS                                               EIG302
C                                                                      EIG302
   43 Q1R=0.                                                           EIG302
```

```
      Q1I=0.
      Q2R=0.
      Q2I=0.
      IF(NUQ-NU)19,21,21
   21 DO 44 J=NU,NUQ
      D1=RTR(J)-XBAR
      D2=RTI(J)-YBAR
      D=D1**2+D2**2
      D1=D1/D
      D2=-D2/D
      Q1R=Q1R+D1
      Q1I=Q1I+D2
      Q2R=Q2R+D1**2-D2**2
   44 Q2I=Q2I+2.*D1*D2

      FIND S1 AND S2

   19 T1R=B(2)/B(1)
      T1I=0.
      T2R=B(3)/B(2)
      T2I=0.
      IF(YBAR)45,46,45
   45 D1=B(1)**2+B(4)**2
      D2=B(2)**2+B(5)**2
      T1R=(B(2)*B(1)+B(5)*B(4))/D1
      T1I=(B(5)*B(1)-B(4)*B(2))/D1
      T2R=(B(3)*B(2)+B(6)*B(5))/D2
      T2I=(B(6)*B(2)-B(5)*B(3))/D2
   46 S1R=T1R+Q1R
      S1I=T1I+Q1I
      S2R=T1R*(T1R-T2R)-T1I*(T1I-T2I)-Q2R
      S2I=T1R*(T1I-T2I)+T1I*(T1R-T2R)-Q2I

      FIND THE NEXT ITERATE

      LLY=LLY+1
      D=ABSF(XBAR)+ABSF(YBAR)
      IF(1.E+7-D*(ABSF(S1R)+ABSF(S1I))) 41,41,42
   41 MARK=1
      GO TO 100
   42 G=N-NUQ
   48 IF(YBAR-ABSF(X))50,50,49
   49 S1I=S1I+1./(2.*YBAR)
      S2R=S2R+1./(4.*YBAR**2)
      G=G-1.
   50 IF(BL1)65,65,66
   65 H=.5*(G-2.)
      GO TO 67
   66 H=G-1.
   67 DR=H*(G*S2R-S1R**2+S1I**2)
      DI=H*(G*S2I-2.*S1R*S1I)
      IF(DI)53,51,53
   51 EI=0.
      ER=SQRTF(ABSF(DR))
      IF(DR)52,54,54
   52 EI=ER
```

```
      ER=0.                                                             EIG302
      GO TO 54                                                          EIG302
   53 CALL CXSQRT(DR,DI,ER,EI)                                          EIG302
   54 IF(S1R*ER+S1I*EI)55,56,56                                         EIG302
   55 ER=-ER                                                            EIG302
      EI=-EI                                                            EIG302
   56 D1=S1R+ER                                                         EIG302
      D2=S1I+EI                                                         EIG302
      D=D1**2+D2**2                                                     EIG302
      X=-G*D1/D                                                         EIG302
      XBAR=XBAR+X                                                       EIG302
      Y=G*D2/D                                                          EIG302
      YBAR=YBAR+Y                                                       EIG302
      DELNEW=ABSF(X)+ABSF(Y)                                            EIG302
      RNEW=DELNEW/DELOLD                                                EIG302
      D=ABSF(XBAR)+ABSF(YBAR)                                           EIG302
                                                                        EIG302
      TEST FOR LINEAR CONVERGENCE                                       EIG302
                                                                        EIG302
      IF(LLY-3)62,62,57                                                 EIG302
57    IF (DELNEW-MAX1F(3.*DELOLD,.5*D))571,571,570                      EIG302
570   IF (BL1) 571,571,572                                              EIG302
572   DELOLD=CAP                                                        EIG302
      ROLD=3.                                                           EIG302
      IF (LLY-15) 14,14,100                                             EIG302
571   IF(RNEW-.7*ROLD) 62,58,58                                         EIG302
   58 MARK=3                                                            EIG302
      IF(DELNEW-.001*EPS*CAP) 70,59,59                                  EIG302
59    IF(BL1)61,61,60                                                   EIG302
60    XBAR=XBAR-X                                                       EIG302
      YBAR=YBAR-Y                                                       EIG302
      BL1=0.                                                            EIG302
      GO TO 48                                                          EIG302
61    BL1=1.                                                            EIG302
      GO TO 63                                                          EIG302
                                                                        EIG302
      TEST FOR AN EIGENVALUE                                            EIG303
                                                                        EIG303
   62 IF(DELNEW-EPS*MAX1F(D,.001*CAP))64,64,63                          EIG303
63    DELOLD=DELNEW                                                     EIG303
      ROLD=RNEW                                                         EIG303
      IF (LLY-15) 23,23,100                                             EIG303
                                                                        EIG303
      DO WE HAVE A COMPLEX APPROACH TO A REAL ROOT                      EIG303
                                                                        EIG303
   64 MARK=2                                                            EIG303
   70 BL1=1                                                             EIG303
      IF(YBAR)71,100,71                                                 EIG303
   71 IF(G2*ABSF(YBAR)-G1)72,100,100                                    EIG303
   72 IF(ONCE)73,73,100                                                 EIG303
   73 X=0.                                                              EIG303
      ONCE=1.                                                           EIG303
      YBAR=0.                                                           EIG303
      GO TO 63                                                          EIG303
                                                                        EIG303
      WE ACCEPT (XBAR,YBAR) AS A ROOT                                   EIG303
```

```
100 NUQ=NUQ+1
    RTR(NUQ)=XBAR
    IF(ABSF(YBAR)-.001*ABSF(XBAR))74,74,75
 74 YBAR=0.
 75 IF(NUQ-NU) 9,76, 9
  9 IF(RTI(NUQ-1))76,76,77
 76 YBAR=ABSF(YBAR)
    RTI(NUQ)=YBAR
    GO TO 78
 77 RTI(NUQ)=-ABSF(YBAR)
 78 WRITE OUTPUT TAPE 6,3,RTR(NUQ),RTI(NUQ),LLY,MARK
  3 FORMAT(11H0EIGENVALUE12X,2E20.8,12X,I3,17H ITERATIONS,TEST I1//)
    LLY=0
    CAP=MAX1F(D,CAP)
    DELOLD=1.
    ROLD=1.
    EGSUM1=EGSUM1+RTR(NUQ)
    EGSUM2=EGSUM2+RTR(NUQ)**2-RTI(NUQ)**2
    IF(NUQ-N1)80,101,101
 80 IF(YBAR)83,84,81
 81 YBAR=-YBAR
    GO TO 23
 83 IF(NUQ-NU)31,84,31
 31 RTI(NUQ-1)=.5*(RTI(NUQ-1)-RTI(NUQ))
    RTI(NUQ)=-RTI(NUQ-1)

    A NEWTON ITERATE TOWARDS NEXT ROOT

 84 ONCE=0.
    Z=0.
    IF((ABSF(Q1R)+ABSF(Q1I))*D-10000.)85,85,14
5   IF(ABSF(EGSUM1-SPUR1)+ABSF(EGSUM2-SPUR2)-1.E-5*CAP)15,15,86
 86 DR=B(3)+2.*(B(2)*Q1R-B(5)*Q1I)
    DI=B(6)+2.*(B(2)*Q1I+B(5)*Q1R)
    D2=DR**2+DI**2
    XBAR=XBAR-2.*(DR*B(2)+DI*B(5))/D2
    YBAR=ABSF(YBAR-2.*(DR*B(5)-DI*B(2))/D2)
    GO TO 23
101 RETURN
  1 FORMAT(1H050X,19HLAGUERRE ITERATIONS/31X,9HREAL PART10X,10HIMAG. P
   1ART22X,1HP11X,7HP PRIME6X,11HP DBL PRIME)
    END
51/5COMPLEX SQUARE ROOT
    SUBROUTINE CXSQRT(A,B,X,Y)
    F=MAX1F(ABSF(A),ABSF(B))
    F=F*SQRTF((A/F)**2+(B/F)**2)
    IF(A)1,1,2
  1 Y=SQRTF((F-A)*.5)
    X=.5*B/Y
    IF(X)4,3,3
  4 X=-X
    Y=-Y
    GO TO 3
  2 X=SQRTF((F+A)*.5)
    Y=.5*B/X
```

```
  3 RETURN
    END
    SUBROUTINE SCALE (D,B,M)
    DIMENSION B(6)
    S = 2. ** (XINTF (LOGF(D)/.69314718056) + 1)
    DO 1 I=1,M
  1 B(I) = B(I)/S
    RETURN
    END
        IDENT      OVFSET
        ENTRY      OVFSET,OVFTST,OVFLU
OVFSET  SLJ        **
        NOP
        CALL       SELECT*
        11         OVFLU
        CALL       ERROR*
        ENA        0
        STA        =SZIG
        SLJ        OVFSET
OVFLU   SLJ        **
        LDA        =D1.
        STA        ZIG
        CALL       SELECT*
        11         OVFLU
        CALL       ERROR*
        SLJ        OVFLU
OVFTST  SLJ        **
        LDA        *
        ARS        24
        SAL        *+2
        INA        1
        SAU        OVFTST
        LDA        ZIG
        STA        **
        ENA        0
        STA        ZIG
        SLJ        OVFTST
        END
    SUBROUTINE MATALG(A,X,NR,NV,IDO,DET,NACT)
    DIMENSION A(NACT,NACT),X(NACT,NACT)
    IF(IDO) 1,2,1
  1 DO 3 I=1,NR
    DO 4 J=1,NR
  4 X(I,J)=0.0
  3 X(I,I)=1.0
    NV=NR
  2 DET=1.0
    NR1=NR-1
    DO 5 K=1,NR1
    IR1=K+1
    PIVOT=0.0
    DO 6 I=K,NR
    Z=ABSF(A(I,K))
    IF(Z-PIVOT) 6,6,7
  7 PIVOT=Z
    IPR=I
```

```
 6 CONTINUE
   IF(PIVOT) 8,9,8
 9 DET=0.0
   RETURN
 8 IF(IPR-K) 10,11,10
10 DO 12 J=K,NR
   Z=A(IPR,J)
   A(IPR,J)=A(K,J)
12 A(K,J)=Z
   DO 13 J=1,NV
   Z=X(IPR,J)
   X(IPR,J)=X(K,J)
13 X(K,J)=Z
   DET=-DET
11 DET=DET*A(K,K)
   PIVOT=1.0/A(K,K)
   DO 14 J=IR1,NR
   A(K,J)=A(K,J)*PIVOT
   DO 14 I=IR1,NR
14 A(I,J)=A(I,J)-A(I,K)*A(K,J)
   DO 5 J=1,NV
   IF(X(K,J)) 15,5,15
15 X(K,J)=X(K,J)*PIVOT
   DO 16 I=IR1,NR
16 X(I,J)=X(I,J)-A(I,K)*X(K,J)
 5 CONTINUE
   IF(A(NR,NR)) 17,9,17
17 DET=DET*A(NR,NR)
   PIVOT=1.0/A(NR,NR)
   DO 18 J=1,NV
   X(NR,J)=X(NR,J)*PIVOT
   DO 18 K=1,NR1
   I=NR-K
   SUM=0.0
   DO 19 L=I,NR1
19 SUM=SUM+A(I,L+1)*X(L+1,J)
18 X(I,J)=X(I,J)-SUM
   END
   SUBROUTINE MATRA(M,N,A,B,MD,ND)
   DIMENSION A(MD,ND),B(MD,ND)
   DO 3 I=1,M
   DO 3 J=1,N
 3 B(I,J)=A(J,I)
   RETURN
   END
   SUBROUTINE MAMUL(M,N,L,A,B,C,MD,ND,LD)
   DIMENSION A(MD,ND),B(ND,LD),C(MD,LD)
   DO 2 I=1,M
   DO 2 J=1,L
   S=0.
   DO 1 K=1,N
 1 S=S+A(I,K)*B(K,J)
 2 C(I,J)=S
   RETURN
   END
   SUBROUTINE MAMUL1(M,N,A,E,B,MD,ND)
```

```
      DIMENSION A(MD,ND),E(ND),B(MD,ND)
      DO 1 I=1,M
      DO 1 J=1,N
    1 B(I,J)=A(I,J)*E(J)
      RETURN
      END
      SUBROUTINE RERTR(N,RTR)
      DIMENSION RTR(25)
      NM1=N-1
      DO 12 I=1,NM1
      IP=I+1
      DO 12 J=IP,N
      IF(RTR(I)-RTR(J)) 12,12,13
   13 TEMP=RTR(I)
      RTR(I)=RTR(J)
      RTR(J)=TEMP
   12 CONTINUE
      RETURN
      END
          END
          FINIS
-EXECUTE,3.
    4
```

APPENDIX 6

PROGRAM REVEIG1

The second program, REVEIG1, was formulated for a maximum 25x25 input based upon equations (7) of the text:

$$UP = (I-K)(I+K)^{-1} \tag{6-1}$$

$$VP = (I+\tilde{K})(I-\tilde{K})^{-1} \tag{6-2}$$

The input is any real square matrix K.

In addition, various operations with UP and VP are printed out for investigation of any obvious relationships, particularly among the diagonal elements; these are $\tilde{V}'V'$, $V'\tilde{V}'$, $\tilde{U}'U'$, $U'\tilde{U}'$, and $\tilde{V}'U'$.

The K outputs of REVEIG were tested as inputs to REVEIG1 in 3x3 and 4x4 matrices, and in both cases the UP and VP compared with the UP and VP in REVEIG. $\tilde{V}'U'$ always resulted in the identity matrix. When K was skew-symmetric, UP and VP were equal as was expected.

```
-COOP,,YOST,S/1S/2S,3,5000.
-FTN,L,E.
      PROGRAM REVEIG1
     0DIMENSION UP(25,25),VP(25,25),FK(25,25),FKT(25,25),VPT(25,25),
     1UPT(25,25),AUX(25,25),AUX1(25,25),T(25,25)
C GIVEN A K MATRIX FIND ITS ASSOCIATED U PRIME MATRIX USING THE FOLLOXING EQUAT
C ON.  UP=(I-K)(I+K) INVERSE
    1 READ 899,N
      IF (N) 2,2,3
    2 STOP
    3 PRINT 870,N
  870 FORMAT (15H1K MATRIX,    N=I3)
  899 FORMAT(I5)
      DO 48 I=1,N
      READ 898,(FK(I,J),J=1,N)
   48 PRINT 898,(FK(I,J),J=1,N)
  898 FORMAT(4E20.10)
C STORE (I+K) IN AUX
      DO 49 I=1,N
      DO 49 J=1,N
      IF(I-J) 490,491,490
  491 AUX(I,I)=FK(I,I)+1.
      GO TO 49
  490 AUX(I,J)=FK(I,J)
   49 CONTINUE
C USE AUX AS TEMPORARY STORAGE
C INVERSE OF (I+K) IN AUX1
      CALL MATALG(AUX,AUX1,N,N,1,DET,25)
C STORE (I-K) IN T
      DO 50 I=1,N
      DO 50 J=1,N
   50 T(I,J)=-FK(I,J)
      DO 51 I=1,N
   51 T(I,I)=T(I,I)+1.
C PREMULTIPLIES (I+K) INVERSE BY (I-K)
      CALL MAMUL(N,N,N,T,AUX1,UP,25,25,25)
      PRINT 897
  897 FORMAT (15H0U-PRIME MATRIX/)
      DO 52 I=1,N
   52 PRINT 896,I,(UP(I,J),J=1,N)
  896 FORMAT(I5/(1X7E17.10))
C TRANSPOSE THE K MATRIX AND FIND ITS ASSOCIATED V MATRIX USING THE FOLLOWING E
C UATION.  VP=(I+KT)(I-KT) INVERSE
      CALL MATRA(N,N,FK,FKT,25,25)
C STORE (I+KT) IN AUX
      DO 53 I=1,N
      DO 53 J=1,N
      AUX(I,J) =FKT(I,J)
C STORE (I-KT) IN AUX1
      AUX1(I,J)=-FKT(I,J)
      IF(I-J) 53,54,53
   54 AUX(I,I) = AUX(I,I)+1.
      AUX1(I,I)=1.-FKT(I,I)
   53 CONTINUE
C STORE INVERSE OF (I-KT) IN T
      CALL MATALG(AUX1,T,N,N,1,DET,25)
```

```
PREMULTIPLY (I-KT) INVERSE BY (I+KT)
    CALL MAMUL(N,N,N,AUX,T,VP,25,25,25)
    PRINT 895
895 FORMAT(15HOV-PRIME MATRIX/)
    DO 55 I=1,N
 55 PRINT 896, I, (VP(I,J), J=1,N)
PRINT OUT ALSO V PRIME TRANSPOSE TIMES V PRIME, V PRIME TIMES V PRIME TRANSP(
E.  SAME FOR U PRIME AND U PRIME TRANSPOSE.
894 FORMAT( 9HOVPT X VP /    )
893 FORMAT( 9HOVP X VPT /    )
892 FORMAT ( 9HOUPT X UP /    )
891 FORMAT ( 9HOUP X UPT /    )
CHECK TO SEE IF V PRIME TRANSPOSE=I WHICH PROVES ORTHONORMALITY.
890 FORMAT (9HOVPT X UP/)
    CALL MATRA (N,N,VP,VPT,25,25)
    CALL MATRA (N,N, UP, UPT, 25,25)
    CALL MAMUL (N,N,N,VPT,VP,T,25,25,25)
    PRINT 894
    ASSIGN 56 TO ISW
    GO TO 60
 56 CALL MAMUL (N,N,N,VP,VPT,T,25,25,25)
    PRINT 893
    ASSIGN 57 TO ISW
    GO TO 60
 57 CALL MAMUL (N,N,N,UPT,UP,T,25,25,25)
    PRINT 892
    ASSIGN 58 TO ISW
    GO TO 60
 58 CALL MAMUL(N,N,N,UP,UPT,T,25,25,25)
    PRINT 891
    ASSIGN 59 TO ISW
    GO TO 60
 59 CALL MAMUL(N,N,N,VPT,UP,T,25,25,25)
    PRINT 890
    ASSIGN 71 TO ISW
 60 DO 61 I=1,N
 61 PRINT 896, I, (T(I,J),J=1,N)
    GO TO ISW
 71 GO TO 1
    END
    SUBROUTINE MATALG(A,X,NR,NV,IDO,DET,NACT)
    DIMENSION A(NACT,NACT),X(NACT,NACT)
    IF(IDO) 1,2,1
  1 DO 3 I=1,NR
    DO 4 J=1,NR
  4 X(I,J)=0.0
  3 X(I,I)=1.0
    NV=NR
  2 DET=1.0
    NR1=NR-1
    DO 5 K=1,NR1
    IR1=K+1
    PIVOT=0.0
    DO 6 I=K,NR
    Z=ABSF(A(I,K))
    IF(Z-PIVOT) 6,6,7
```

```
 7 PIVOT=Z
   IPR=I
 6 CONTINUE
   IF(PIVOT) 8,9,8
 9 DET=0.0
   RETURN
 8 IF(IPR-K) 10,11,10
10 DO 12 J=K,NR
   Z=A(IPR,J)
   A(IPR,J)=A(K,J)
12 A(K,J)=Z
   DO 13 J=1,NV
   Z=X(IPR,J)
   X(IPR,J)=X(K,J)
13 X(K,J)=Z
   DET=-DET
11 DET=DET*A(K,K)
   PIVOT=1.0/A(K,K)
   DO 14 J=IR1,NR
   A(K,J)=A(K,J)*PIVOT
   DO 14 I=IR1,NR
14 A(I,J)=A(I,J)-A(I,K)*A(K,J)
   DO 5 J=1,NV
   IF(X(K,J)) 15,5,15
15 X(K,J)=X(K,J)*PIVOT
   DO 16 I=IR1,NR
16 X(I,J)=X(I,J)-A(I,K)*X(K,J)
 5 CONTINUE
   IF(A(NR,NR)) 17,9,17
17 DET=DET*A(NR,NR)
   PIVOT=1.0/A(NR,NR)
   DO 18 J=1,NV
   X(NR,J)=X(NR,J)*PIVOT
   DO 18 K=1,NR1
   I=NR-K
   SUM=0.0
   DO 19 L=I,NR1
19 SUM=SUM+A(I,L+1)*X(L+1,J)
18 X(I,J)=X(I,J)-SUM
   END
   SUBROUTINE MATRA(M,N,A,B,MD,ND)
   DIMENSION A(MD,ND),B(MD,ND)
   DO 3 I=1,M
   DO 3 J=1,N
 3 B(I,J)=A(J,I)
   RETURN
   END
   SUBROUTINE MAMUL(M,N,L,A,B,C,MD,ND,LD)
   DIMENSION A(MD,ND),B(ND,LD),C(MD,LD)
   DO 2 I=1,M
   DO 2 J=1,L
   S=0.
   DO 1 K=1,N
 1 S=S+A(I,K)*B(K,J)
 2 C(I,J)=S
   RETURN
```

```
      END
         END
         FINIS
-EXECUTE,3.
   4
  0.0000000000E 00    2.0000000000E 00    5.0000000000E 00    1.0000000000E
  1.0000000000E 00    0.0000000000E 00    2.0000000000E 00    2.0000000000E
  4.0000000000E 00    3.0000000000E 00    0.0000000000E 00    3.0000000000E
  5.0000000000E 00    4.0000000000E 00    3.0000000000E 00    0.0000000000E
```

APPENDIX 7

PROGRAM REVEIG2

A short third program, REVEIG2, was formulated to carry out the calculation in equation (3-5) of Appendix 3, and to compute the corresponding matrix K(U') from equation (8) of the text. The program printout notation requires special description however. The input to REVEIG2 was an arbitrary real diagonal matrix, G, and an orthonormal matrix X'*, as in equation (3-5). The latter matrix was, however, the result obtained from REVEIG for a symmetric matrix, and called U' or UP in REVEIG instead of X'*. It is therefore called UP in REVEIG2, also, necessitating the introduction of the program description UPP, or U'', for the calculation in equation (3-5). Thus in REVEIG2 the following is calculated

$$UPP = G(UP) \tag{7-1}$$

and

$$K = (I+UPP)^{-1}(I-UPP) \tag{7-2}$$

```
-COOP,,YOST,S/1S/2S,3,5000.
-FTN,L,E.
      PROGRAM REVE1G2
      DIMENSION UP(25,25),G(25), AUX(25,25),UPP(25,25),FK(25,25)
      DIMENSION AUX1(25,25),AUX2(25,25)
C FIND THE K MATRIX DERIVED FROM UPP=G*UP WHERE G=B TO THE MINUS ONE-HALF
C AND K=(I+UPP)INVERSE TIMES (I-UPP)
    1 READ 899,N
      IF(N) 2,2,3
    2 STOP
    3 PRINT 898,N
  898 FORMAT(16H1UP MATRIX,    N=I3/)
  899 FORMAT(I5)
      DO 48 I=1,N
      READ 897,(UP(I,J),J=1,N)
   48 PRINT 897,(UP(I,J),J=1,N)
  897 FORMAT(4E20.10)
      PRINT 896,N
  896 FORMAT(15H0G MATRIX,    N=I3/)
      READ 895,(G( J ),J=1,N)
   47 PRINT 895,(G(  J),J=1,N)
  895 FORMAT (4E20.10)
C
      DO 46 I = 1,N
      DO 46 J = 1,N
   46 UPP(I,J) = UP(I,J) * G(I)
      PRINT 894
      DO 45 I = 1,N
   45 PRINT 893,I, (UPP(I,J),J=1,N)
  894 FORMAT  (11H0UPP MATRIX/)
  893 FORMAT (I5/(1X7E17.10))
C
      DO 44 I = 1,N
      DO 44 J = 1,N
      IF(I-J) 43,42,43
   42 AUX(I,J)= UPP(I,J) + 1.
      AUX2(I,J)=1.-UPP(I,J)
      GO TO 44
   43 AUX(I,J) = UPP(I,J)
      AUX2(I,J)=UPP(I,J)
   44 CONTINUE
      CALL MATALG (AUX,AUX1,N,N,1,DET,25)
      CALL MAMUL (N,N,N,AUX1,AUX2,FK,25,25,25)
C
      PRINT 892
  892 FORMAT(9H0K MATRIX/)
      DO 40 I = 1,N
   40 PRINT 891, I, (FK(I,J), J=1,N)
  891 FORMAT (I5/(1X7E17.10))
C
      DO 39 I=1,N
   39 G(I)=1./G(I)**2
      PRINT 890, (G(I), I=1,N)
  890 FORMAT ( 9H0B MATRIX /(1X7E17.10))
      GO TO 1
      END
```

```
      SUBROUTINE MATALG(A,X,NR,NV,IDO,DET,NACT)
      DIMENSION A(NACT,NACT),X(NACT,NACT)
      IF(IDO) 1,2,1
    1 DO 3 I=1,NR
      DO 4 J=1,NR
    4 X(I,J)=0.0
    3 X(I,I)=1.0
      NV=NR
    2 DET=1.0
      NR1=NR-1
      DO 5 K=1,NR1
      IR1=K+1
      PIVOT=0.0
      DO 6 I=K,NR
      Z=ABSF(A(I,K))
      IF(Z-PIVOT) 6,6,7
    7 PIVOT=Z
      IPR=I
    6 CONTINUE
      IF(PIVOT) 8,9,8
    9 DET=0.0
      RETURN
    8 IF(IPR-K) 10,11,10
   10 DO 12 J=K,NR
      Z=A(IPR,J)
      A(IPR,J)=A(K,J)
   12 A(K,J)=Z
      DO 13 J=1,NV
      Z=X(IPR,J)
      X(IPR,J)=X(K,J)
   13 X(K,J)=Z
      DET=-DET
   11 DET=DET*A(K,K)
      PIVOT=1.0/A(K,K)
      DO 14 J=IR1,NR
      A(K,J)=A(K,J)*PIVOT
      DO 14 I=IR1,NR
   14 A(I,J)=A(I,J)-A(I,K)*A(K,J)
      DO 5 J=1,NV
      IF(X(K,J)) 15,5,15
   15 X(K,J)=X(K,J)*PIVOT
      DO 16 I=IR1,NR
   16 X(I,J)=X(I,J)-A(I,K)*X(K,J)
    5 CONTINUE
      IF(A(NR,NR)) 17,9,17
   17 DET=DET*A(NR,NR)
      PIVOT=1.0/A(NR,NR)
      DO 18 J=1,NV
      X(NR,J)=X(NR,J)*PIVOT
      DO 18 K=1,NR1
      I=NR-K
      SUM=0.0
      DO 19 L=I,NR1
   19 SUM=SUM+A(I,L+1)*X(L+1,J)
   18 X(I,J)=X(I,J)-SUM
      END
```

```
      SUBROUTINE MAMUL(M,N,L,A,B,C,MD,ND,LD)
      DIMENSION A(MD,ND),B(ND,LD),C(MD,LD)
      DO 2 I=1,M
      DO 2 J=1,L
      S=0.
      DO 1 K=1,N
    1 S=S+A(I,K)*B(K,J)
    2 C(I,J)=S
      RETURN
      END
          END
          FINIS
EXECUTE.
    4
    9.4067179924E-02   -2.3754675249E-01    3.7708598062E-01   -8.9024101645E-
    3.1644217545E-01   -6.2782339524E-01    5.6007619310E-01    4.3819719533E-
    5.5556147215E-01   -4.3075473215E-01   -7.0046990430E-01   -1.2305981130E-
    7.6312984866E-01    6.0320782940E-01    2.3122009925E-01    1.7619122150E-
    2.0000000000E 00    4.0000000000E 00    6.0000000000E 00    8.0000000000E
```

APPENDIX 8

NUMERICAL RESULTS

The following 3 x 3 unsymmetric matrix was tested in REVEIG so that the results could be compared with the known eigenvalues and eigenvectors. This example was taken from Lanczos [2].

$$A = \begin{bmatrix} 33 & 16 & 72 \\ -24 & -10 & -57 \\ -8 & -4 & -17 \end{bmatrix}$$

Complete Eigenvalue Analysis

$\lambda = 1$	$\lambda = 2$	$\lambda = 3$	$\lambda = 1$	$\lambda = 2$	$\lambda = 3$
u_1	u_2	u_3	v_1	v_2	v_3
-15/4	-4	-4	1/4	0	4/3
3	13/4	3	0	-1/3	4/3
1	1	1	1	1	1

The computer results compare exactly with the above answers as shown in the accompanying data sheets.

The following 4 x 4 symmetric matrix was tested in REVEIG so that the results could be compared with its known eigenvalues and eigenvectors.

$$A = \begin{bmatrix} 2.8 & -0.8 & 0.0 & 0.0 \\ -0.8 & 1.4 & -0.6 & 0.0 \\ 0.0 & -0.6 & 1.0 & -0.4 \\ 0.0 & 0.0 & -0.4 & 0.4 \end{bmatrix}$$

$\lambda_1 = .108$	$\lambda_2 = .685$	$\lambda_3 = 1.612$	$\lambda_4 = 3.194$
x_1	x_2	x_3	x_4
.0940	-.2375	.3771	-.8902
.3164	-.6278	.5601	.4382
.5556	-.4308	-.7005	-.1231
.7631	.6302	.2312	.0176

An additional 3 x 3 symmetric A was tested in order to verify the program and the theory for 3 x 3. These results are also shown, and they contain the same UP and VP as in REVEIG1 for the same case.

```
52553 Q1Q10130                              52567 Q1Q1033C
52513 Q3Q10040                              52513 Q3Q10140
52513 Q3Q10340                              52513 Q3Q1044C
52743 Q7QLODRC                              52753 Q7QLODDC
52771 Q7QLODLC                              53006 Q7QLODID
52736 Q7QFLOAT                              52733 Q7QLDIC2
52756 Q7QLDDC3                              52765 Q7QLDCC3
53010 Q7QLDID2                              53017 Q7QLDRD2
47514 SCALE                                 47557 CXSQRT
47650 Q8QXMODF                              47456 XTOI
47455 Q8QXTOI                               47377 LOGF
47364 Q2QLOADA                              47347 OVFLU

                                             TRACE =  6.00000000E 00

                                             ALMOST TRIANGULAR FORM

                                             TRACE =  6.00000000E 00

                                              LAGUERRE ITERATIONS
                         REAL PART           IMAG. PART
ITERATE                3.15470054E 00                    0
ITERATE                3.00016635E 00                    0

EIGENVALUE             3.00000001E 00                    0

ITERATE                2.33311156E 00                    0
ITERATE                1.99999999E 00                    0

EIGENVALUE             1.99999999E 00                    0

ITERATE                1.00000000E 00                    0

EIGENVALUE             1.00000000E 00                    0

                              SUM OF EIGENVALUES =  6.00000000E 00
@U MATRIX

   1
-3.7500000003E 00-3.9999999966E 00-4.0000000104E 00   ← ORTHOGONAL
   2
 3.0000000007E 00 3.2499999950E 00 3.0000000176E 00
   3
 1.0000000000E 00 1.0000000000E 00 1.0000000000E 00
                                             TRACE =  6.00000000E 00

                                             ALMOST TRIANGULAR FORM

                                             TRACE =  6.00000000E 00

EIGENVALUE             3.00000000E 00                    0

EIGENVALUE             1.00000000E 00                    0

EIGENVALUE             2.00000000E 00                    0

                              SUM OF EIGENVALUES =  6.00000000E 00
@V MATRIX

   1
 2.4999999999E-01-1.4551915228E-10 1.3333333321E 00
   2
                0-3.3333333353E-01 1.3333333318E 00   ← ORTHOGONAL
   3
 1.0000000000E 00 1.0000000000E 00 1.0000000000E 00
@W MATRIX

   1
 6.2499999913E-02 8.4401108322E-10-2.6193447411E-09
   2
-2.9103830456E-10-8.3333331742E-02-5.8789737523E-09   ← ORTHOGONAL
   3
 4.6566128730E-10-2.0954757929E-09-3.3333332324E-01
@U MATRIX

   1
-3.7500000003E 00 3.9999999966E 00 4.0000000104E 00
   2
 3.0000000007E 00-3.2499999950E 00-3.0000000176E 00
```

```
 1.0000000000E 00-1.0000000000E 00-1.0000000000E 00
CW MATRIX

  1
 6.2499999913E-02-8.4401108322E-10 2.6193447411E-09
  2
-2.9103830456E-10 8.3333331742E-02 5.8789737523E-09
  3
 4.6566128730E-10 2.0954757929E-09 3.3333332324E-01

U-PRIME MATRIX

  1
-1.5000000012E 01 1.3856406581E 01 6.9282033532E 00
  2
 1.2000000011E 01-1.1258330339E 01-5.1961525318E 00 ← ORTHONORMAL
  3
 4.0000000028E 00-3.4641016482E 00-1.7320508338E 00

V-PRIME MATRIX

  1
 1.0000000007E 00-5.0409313526E-10 2.3094011096E 00
  2
                 0-1.1547005501E 00 2.3094011091E 00 ← ORTONORMAL
  3
 4.0000000028E 00 3.4641016482E 00 1.7320508338E 00

V-PRIME-TRANSPOSE TIMES U-PRIME

  1
 9.9999999951E-01-1.2107193470E-08 1.7927959561E-08
  2
-3.7252902985E-09 9.9999999951E-01 3.5041011870E-08  = I
  3
 3.2596290112E-09 1.2922100723E-08 1.0000000008E 00

K MATRIX

  1
-9.2698519866E-02 1.1984252959E 00 8.0279293237E-02
  2
 1.0378667465E 00 5.1045085593E-01-8.9881895536E-01
  3
 4.6349262993E-02-5.9921264054E-01 9.5986034195E-01

KT MATRIX

  1
-9.2698525495E-02 1.0378667592E 00 4.6349262997E-02
  2
 1.1984252946E 00 5.1045087465E-01-5.9921264714E-01
  3
 8.0279277416E-02-8.9881896684E-01 9.5986036127E-01

                                              TRACE =  5.60000000E 00

                                              ALMOST TRIANGULAR FORM

                                              TRACE =  5.60000000E 00

                                               LAGUERRE ITERATIONS
                             REAL PART          IMAG. PART
ITERATE                     3.41990099E 00           0
ITERATE                     3.19399497E 00           0

EIGENVALUE                  3.19377848E 00           0

ITERATE                     2.45552682E 00           0
ITERATE                     1.61669232E 00           0
ITERATE                     1.61178040E 00           0

EIGENVALUE                  1.61178040E 00           0

ITERATE                     1.03874635E 00           0
ITERATE                     6.85642675E-01           0

EIGENVALUE                  6.85642675E-01           0

ITERATE                     1.08798451E-01           0

EIGENVALUE                  1.08798451E-01           0
```

SUM OF EIGENVALUES = 5.60000000E 00

U MATRIX

```
1  1.2326497265E-01-3.9380581770E-01 1.6308529485E 00-5.0526979089E 01
2  4.1466360672E-01-1.0408077692E 00 2.4222643054E 00 2.4870546421E 01
3  7.2800385559E-01-7.1410666635E-01-3.0294507551E 00-6.9844462315E 00
4  1.0000000000E 00 1.0000000000E 00 1.0000000000E 00 1.0000000000E 00
```

TRACE = 5.60000000E 00

ALMOST TRIANGULAR FORM

TRACE = 5.60000000E 00

LAGUERRE ITERATIONS

	REAL PART	IMAG. PART
ITERATE	3.41990099E 00	0
ITERATE	3.19399497E 00	0
EIGENVALUE	3.19377848E 00	0
ITERATE	2.45552682E 00	0
ITERATE	1.61669232E 00	0
ITERATE	1.61178040E 00	0
EIGENVALUE	1.61178040E 00	0
ITERATE	1.03874635E 00	0
ITERATE	6.85642675E-01	0
EIGENVALUE	6.85642675E-01	0
ITERATE	1.08798451E-01	0
EIGENVALUE	1.08798451E-01	0

SUM OF EIGENVALUES = 5.60000000E 00

V MATRIX

```
1  1.2326497265E-01-3.9380581770E-01 1.6308529485E 00-5.0526979089E 01
2  4.1466360672E-01-1.0408077692E 00 2.4222643054E 00 2.4870546421E 01
3  7.2800385559E-01-7.1410666635E-01-3.0294507551E 00-6.9844462315E 00
4  1.0000000000E 00 1.0000000000E 00 1.0000000000E 00 1.0000000000E 00
```

W MATRIX

```
1  1.7171297740E 00 2.6746420190E-08 6.7346263677E-08-1.0477378964E-09
2  2.6746420190E-08 2.7483121654E 00 9.2666596174E-08-7.5669959188E-09
3  6.7346263677E-08 9.2666596174E-08 1.8704617582E 01-6.5192580223E-08
4 -1.0477378964E-09-7.5669959188E-09-6.5192580223E-08 3.2213021844E 03
```

U-PRIME MATRIX

```
1  9.4067179924E-02-2.3754675249E-01 3.7708598062E-01-8.9024101645E-01
2  3.1644217547E-01-6.2782339524E-01 5.6007619310E-01 4.3819719533E-01
3  5.5556147215E-01-4.3075473215E-01-7.0046990430E-01-1.2305981130E-01
4  7.6312984866E-01 6.0320782940E-01 2.3122009925E-01 1.7619122150E-02
```

V-PRIME MATRIX

```
1  9.4067179924E-02-2.3754675249E-01 3.7708598062E-01-8.9024101645E-01
2  3.1644217545E-01-6.2782339524E-01 5.6007619310E-01 4.3819719533E-01
3  5.5556147215E-01-4.3075473215E-01-7.0046990430E-01-1.2305981130E-01
```

V-PRIME-TRANSPOSE TIMES U-PRIME

```
 1
 1.0000000000E 00 1.2340024114E-08 1.1888914742E-08-1.0913936421E-11
 2
 1.2340024114E-08 9.9999999995E-01 1.2933014659E-08-7.9580786406E-11
 3
 1.1888914742E-08 1.2933014659E-08 1.0000000000E 00-2.6886937121E-10
 4
-1.0913936421E-11-7.9580786406E-11-2.6886937121E-10 1.0000000000E 00
```

= I

K MATRIX

```
 1
-4.0017766878E-09-1.4996701786E 00 3.3970411263E-01 1.5616807591E 00
 2
 1.4996701212E 00 1.8728314899E-08-4.0592364591E 00 3.9046217602E-01
 3
-3.3970411444E-01 4.0592365213E 00 3.2716343412E-08-1.9242020291E 00
 4
-1.5616807415E 00-3.9046223382E-01 1.9242020085E 00 1.8699211068E-08
```

KT MATRIX

```
 1
 4.0017766878E-09 1.4996701786E 00-3.3970411263E-01-1.5616807591E 00
 2
-1.4996701212E 00-1.8728314899E-08 4.0592364591E 00-3.9046217602E-01
 3
 3.3970411444E-01-4.0592365213E 00-3.2716343412E-08 1.9242020291E 00
 4
 1.5616807415E 00 3.9046223382E-01-1.9242020085E 00-1.8699211068E-08
```

```
                                        SUM OF EIGENVALUES =  5.20000000E 00
U MATRIX

 1
 3.0772417209E-01-5.8228119506E-01 7.4412236765E 00
 2
 9.0012116814E-01-9.1189723152E-01-3.6548905973E 00
 3
 1.0000000000E 00 1.0000000000E 00 1.0000000000E 00
                                                      TRACE =  5.20000000E 00
                                                      ALMOST TRIANGULAR FORM
                                                      TRACE =  5.20000000E 00
                                                        LAGUERRE ITERATIONS
                                 REAL PART             IMAG.. PART
ITERATE                         3.32218337E 00                    0
ITERATE                         3.19296062E 00                    0

EIGENVALUE                      3.19293436E 00                    0

ITERATE                         2.16568534E 00                    0
ITERATE                         1.54713834E 00                    0

EIGENVALUE                      1.54713834E 00                    0

ITERATE                         7.69321492E-01                    0
ITERATE                         4.59927299E-01                    0

EIGENVALUE                      4.59927299E-01                    0

                                        SUM OF EIGENVALUES =  5.20000000E 00
V MATRIX

 1
 3.0772417209E-01-5.8228119506E-01 7.4412236765E 00
 2
 9.0012116814E-01-9.1189723152E-01-3.6548905973E 00
 3
 1.0000000000E 00 1.0000000000E 00 1.0000000000E 00
W MATRIX

 1
 1.9049122834E 00 2.9103830457E-11 1.3387762010E-09
 2
 2.9103830457E-11 2.1706079510E 00 2.0954757929E-09
 3
 1.3387762010E-09 2.0954757929E-09 6.9730035083E 01
U-PRIME MATRIX

 1
 2.2295854460E-01-3.9522287214E-01 8.9111636088E-01
 2
 6.5217400456E-01-6.1894948008E-01-4.3768779841E-01
 3
 7.2454023707E-01 6.7874916019E-01 1.1975400816E-01
V-PRIME MATRIX                                              VP = UP

 1
 2.2295854460E-01-3.9522287214E-01 8.9111636088E-01
 2
 6.5217400456E-01-6.1894948008E-01-4.3768779841E-01
 3
 7.2454023707E-01 6.7874916019E-01 1.1975400816E-01
V-PRIME-TRANSPOSE TIMES U-PRIME

 1
 9.9999999998E-01 3.6379788071E-11 1.1641532182E-10
 2
 3.6379788071E-11 1.0000000000E 00 1.7462298274E-10      = I
 3
 1.1641532182E-10 1.7462298274E-10 1.0000000000E 00
```

```
 1
-1.2732925825E-11 1.4471543472E 00-2.3015283608E-01
 2
-1.4471543472E 00 2.7648638933E-10 1.5425447924E 00
 3
 2.3015283620E-01-1.5425447925E 00-1.6734702512E-10
```

KT MATRIX

```
 1
 1.2732925825E-11-1.4471543472E 00 2.3015283608E-01
 2
 1.4471543472E 00-2.7648638933E-10-1.5425447924E 00
 3
-2.3015283620E-01 1.5425447925E 00 1.6734702512E-10
```

The K matrix derived from REVEIG using a 4 x 4 symmetric A was used as input into REVEIG1. The results for UP and VP from REVEIG1 compared with those of REVEIG. In addition, UP equalled VP as required in this case.

Other sample results are included, as discussed in Section 4 of the text.

```
K MATRIX,    N=  4  = SKEW-SYMMETRIC
  -4.0017766878E-09    -1.4996701786E 00     3.3970411263E-01     1.561680759
   1.4996701212E 00     1.8728314899E-08    -4.0592364591E 00     3.904621760
  -3.3970411444E-01     4.0592365213E 00     3.2716343412E-08    -1.924202029
  -1.5616807415E 00    -3.9046223382E-01     1.9242020085E 00     1.869921106

U-PRIME MATRIX  = V PRIME

   1
 9.4067179903E-02-2.3754675252E-01 3.7708598059E-01-8.9024101645E-01
   2
 3.1644217541E-01-6.2782339526E-01 5.6007619314E-01 4.3819719532E-01
   3
 5.5556147218E-01-4.3075473213E-01-7.0046990433E-01-1.2305981130E-01
   4
 7.6312984867E-01 6.0320782942E-01 2.3122009924E-01 1.7619122184E-02

V-PRIME MATRIX = U PRIME

   1
 9.4067178376E-02-2.3754675862E-01 3.7708598232E-01-8.9024101633E-01
   2
 3.1644217654E-01-6.2782340635E-01 5.6007619757E-01 4.3819719545E-01
   3
 5.5556148578E-01-4.3075472992E-01-7.0046990532E-01-1.2305981148E-01
   4
 7.6312983852E-01 6.0320781703E-01 2.3122008239E-01 1.7619122242E-02

VPT X VP

   1
 1.0000000001E 00-1.2267264538E-08-1.1859810911E-08 8.1854523157E-12
   2
-1.2267264538E-08 1.0000000000E 00-1.2958480511E-08 3.8198777474E-11
   3
-1.1859810911E-08-1.2958480511E-08 9.9999999993E-01 2.5011104298E-10
   4
 8.1854523157E-12 3.8198777474E-11 2.5011104298E-10 9.9999999998E-01

VP X VPT

   1
 1.0000000018E 00 4.3291947804E-09 5.3296389523E-10-4.4537955546E-09
   2
 4.3291947804E-09 1.0000000099E 00 2.2491803975E-09-4.8448782763E-09
   3
 5.3296389523E-10 2.2491803975E-09 1.0000000073E 00 1.1539555089E-08
   4
-4.4537955546E-09-4.8448782763E-09 1.1539555089E-08 9.9999998095E-01

UPT X UP

   1
 1.0000000001E 00 1.2369127944E-08 1.1797965271E-08 5.9117155614E-12
   2
 1.2369127944E-08 1.0000000000E 00 1.2914824765E-08-4.0927261580E-11
   3
 1.1797965271E-08 1.2914824765E-08 1.0000000001E 00-2.1941559680E-10
   4
 5.9117155614E-12-4.0927261580E-11-2.1941559680E-10 1.0000000000E 00

UP X UPT

   1
 9.9999999811E-01-4.3364707381E-09-5.4933479986E-10 4.4110493036E-09
   2
-4.3364707381E-09 9.9999999013E-01-2.3073880584E-09 4.8153196985E-09
   3
-5.4933479986E-10-2.3073880584E-09 9.9999999269E-01-1.1431325220E-08
   4
 4.4110493036E-09 4.8153196985E-09-1.1431325220E-08 1.0000000192E 00

VPT X UP

   1
 1.0000000001E 00 4.3655745685E-11-1.0913936421E-11 9.0949470175E-12
   2
 5.8207660913E-11 1.0000000000E 00-2.5465851650E-11 3.7744030124E-11   = I
   3
-4.7293724492E-11-3.2741809264E-11 1.0000000000E 00 5.6843418860E-12
   4
 8.6401996666E-12-3.0922819860E-11 2.5011104299E-11 9.9999999998E-01
```

```
K MATRIX,    N=  3 = SKEW-SYMMETRIC
  -1.2732925825E-11     1.4471543472E 00    -2.3015283608E-01
  -1.4471543472E 00     2.7648638933E-10     1.5425447924E 00
   2.3015283620E-01    -1.5425447925E 00    -1.6734702512E-10

U-PRIME MATRIX = V PRIME

   1
 2.2295854460E-01-3.9522287215E-01 8.9111636085E-01
   2
 6.5217400455E-01-6.1894948013E-01-4.3768779843E-01
   3
 7.2454023702E-01 6.7874916019E-01 1.1975400812E-01
V-PRIME MATRIX = U PRIME

   1
 2.2295854451E-01-3.9522287225E-01 8.9111636090E-01
   2
 6.5217400463E-01-6.1894947997E-01-4.3768779839E-01
   3
 7.2454023706E-01 6.7874916012E-01 1.1975400796E-01
VPT X VP

   1
 1.0000000000E 00-1.4551915228E-11-1.2732925825E-10
   2
-1.4551915228E-11 9.9999999988E-01-1.4006218407E-10
   3
-1.2732925825E-10-1.4006218407E-10 1.0000000000E 00
VP X VPT

   1
 1.0000000000E 00-2.1827872843E-11-1.2369127944E-10
   2
-2.1827872843E-11 9.9999999998E-01 1.3733369996E-10
   3
-1.2369127944E-10 1.3733369996E-10 9.9999999991E-01
UPT X UP

   1
 9.9999999991E-01-2.9103830457E-11 7.8216544352E-11
   2
-2.9103830457E-11 1.0000000001E 00 1.7462298274E-10
   3
 7.8216544352E-11 1.7462298274E-10 9.9999999993E-01
UP X UPT

   1
 9.9999999988E-01 7.2759576141E-12 1.0913936421E-10
   2
 7.2759576141E-12 1.0000000001E 00-1.6916601453E-10
   3
 1.0913936421E-10-1.6916601453E-10 1.0000000000E 00
VPT X UP

   1
 9.9999999995E-01-3.6379788071E-11-3.4560798667E-11
   2
-7.2759576141E-12 9.9999999998E-01 5.4569682105E-12   = I
   3
-7.2759576141E-12 2.9103830457E-11 9.9999999998E-01
```

```
K MATRIX,    N=  4 = SKEW-SYMMETRIC
                    0     -1.0000000000E 00    -4.0000000000E 00    -5.000000000
   1.0000000000E 00                    0    -3.0000000000E 00    -4.000000000
   4.0000000000E 00     3.0000000000E 00                    0    -3.000000000
   5.0000000000E 00     4.0000000000E 00     3.0000000000E 00

U-PRIME MATRIX = V PRIME

   1
-1.3580246915E-01-6.1728395055E-01-3.9506172836E-01 6.6666666665E-01
   2
-9.6296296294E-01 2.5925925934E-01-7.4074074102E-02                0
   3
-1.9753086417E-01-7.1604938272E-01 6.1728394998E-02-6.6666666665E-01
   4
 1.2345679009E-01 1.9753086422E-01-9.1358024699E-01-3.3333333330E-01

V-PRIME MATRIX = U PRIME

   1
-1.3580246915E-01-6.1728395055E-01-3.9506172836E-01 6.6666666665E-01
   2
-9.6296296294E-01 2.5925925934E-01-7.4074074102E-02                0
   3
-1.9753086417E-01-7.1604938272E-01 6.1728394998E-02-6.6666666665E-01
   4
 1.2345679009E-01 1.9753086422E-01-9.1358024699E-01-3.3333333330E-01

VPT X VP

   1
 9.9999999993E-01-9.4587448984E-11 5.4569682106E-11-1.7280399334E-11
   2
-9.4587448984E-11 9.9999999998E-01-4.7293724492E-11 4.1836756282E-11
   3
 5.4569682106E-11-4.7293724492E-11 1.0000000001E 00 6.5483618528E-11
   4
-1.7280399334E-11 4.1836756282E-11 6.5483618528E-11 9.9999999995E-01

VP X VPT

   1
 9.9999999988E-01-1.6370904632E-11-7.2759576141E-12 3.6379788071E-11
   2
-1.6370904632E-11 1.0000000000E 00-8.6060936155E-11 8.5492501967E-11
   3
-7.2759576141E-12-8.6060936155E-11 9.9999999998E-01 1.4551915228E-11
   4
 3.6379788071E-11 8.5492501967E-11 1.4551915228E-11 1.0000000001E 00

UPT X UP

   1
 9.9999999993E-01-9.4587448984E-11 5.4569682106E-11-1.7280399334E-11
   2
-9.4587448984E-11 9.9999999998E-01-4.7293724492E-11 4.1836756282E-11
   3
 5.4569682106E-11-4.7293724492E-11 1.0000000001E 00 6.5483618528E-11
   4
-1.7280399334E-11 4.1836756282E-11 6.5483618528E-11 9.9999999995E-01

UP X UPT

   1
 9.9999999988E-01-1.6370904632E-11-7.2759576141E-12 3.6379788071E-11
   2
-1.6370904632E-11 1.0000000000E 00-8.6060936155E-11 8.5492501967E-11
   3
-7.2759576141E-12-8.6060936155E-11 9.9999999998E-01 1.4551915228E-11
   4
 3.6379788071E-11 8.5492501967E-11 1.4551915228E-11 1.0000000001E 00

VPT X UP

   1
 9.9999999993E-01-9.4587448984E-11 5.4569682106E-11-1.7280399334E-11
   2
-9.4587448984E-11 9.9999999998E-01-4.7293724492E-11 4.1836756282E-11   = I
   3
 5.4569682106E-11-4.7293724492E-11 1.0000000001E 00 6.5483618528E-11
   4
-1.7280399334E-11 4.1836756282E-11 6.5483618528E-11 9.9999999995E-01
```

```
K MATRIX,    N=  4  = UNSYMMETRIC
   9.0000000000E 00    2.0000000000E 00    5.0000000000E 00    1.000000000
   1.0000000000E 00    6.0000000000E 00    2.0000000000E 00    2.000000000
   4.0000000000E 00    3.0000000000E 00    7.0000000000E 00    3.000000000
   5.0000000000E 00    4.0000000000E 00    3.0000000000E 00    4.000000000

U-PRIME MATRIX

     1
-7.6344086018E-01-3.4408602157E-02-1.6344086019E-01 6.4516129030E-02
     2
 5.0179211473E-02-6.3154121863E-01-8.3154121875E-02-1.0752688172E-01
     3
-4.3010752691E-02-3.0107526872E-02-6.4301075265E-01-1.9354838712E-01
     4
-2.5089605734E-01-2.4229390683E-01 1.5770609340E-02-4.6236559140E-01

V-PRIME MATRIX

     1
-1.2068965517E 00-2.0689655175E-01-1.3793103452E-01 7.5862068973E-01
     2
 1.7733990149E-01-1.6798029557E 00-1.6748768472E-01 7.7832512329E-01
     3
 2.7586206894E-01 2.7586206903E-01-1.4827586206E 00-3.4482758626E-01
     4
-3.2512315277E-01 2.4630541874E-01 6.4039408874E-01-2.0935960592E 00

VPT X VP

     1
 1.6698536727E 00-5.2173068997E-02-4.8047756554E-01-1.9199689376E-01
     2
-5.2173068997E-02 3.0013103936E 00 5.8579436492E-02-2.0751777529E 00
     3
-4.8047756554E-01 5.8579436492E-02 2.6557548108E 00-1.0644276738E 00
     4
-1.9199689376E-01-2.0751777529E 00-1.0644276738E 00 5.6833458720E 00

VP X VPT

     1
 2.0939357908E 00 7.4706981507E-01-4.4708680144E-01-1.3351452355E 00
     2
 7.4706981507E-01 3.4870295327E 00-4.3451673200E-01-2.2081584125E 00
     3
-4.4708680144E-01-4.3451673200E-01 2.4696789536E 00-2.4936300312E-01
     4
-1.3351452355E 00-2.2081584125E 00-2.4936300312E-01 4.9596204719E 00

UPT X UP

     1
 6.5015865669E-01 5.6664225799E-02 1.4430441538E-01 6.9680502568E-02
     2
 5.6664225799E-02 4.5964106319E-01 7.3677367961E-02 1.8354337690E-01
     3
 1.4430441538E-01 7.3677367961E-02 4.4733906290E-01 1.1555863876E-01
     4
 6.9680502568E-02 1.8354337690E-01 1.1555863876E-01 2.6696727946E-01

UP X UPT
```

```
 6.1490114459E-01-9.9248468005E-03 1.2647936175E-01 1.6747369637E-01
    2
-9.9248468005E-03 4.1983890238E-01 9.1136547578E-02 1.8883416196E-01
    3
 1.2647936175E-01 9.1136547578E-02 4.5368019420E-01 9.7435541679E-02
    4
 1.6747369637E-01 1.8883416196E-01 9.7435541679E-02 3.3568582111E-01

VPT X UP

    1
 9.9999999995E-01 2.3646862246E-11-3.0468072509E-11 2.9103830457E-11
    2
-1.8189894035E-12 1.0000000000E 00-3.2969182939E-12-2.7284841053E-11
    3
 1.0913936421E-11-5.4569682106E-11 9.9999999993E-01-2.9103830457E-11
    4
-1.4551915228E-11-2.9103830457E-11-3.6379788071E-11 1.0000000001E 00
```

= I

```
K MATRIX,    N=  4 = UNSYMMETRIC WITH ZERO DIAGONAL
                0    2.0000000000E 00    5.0000000000E 00    1.0000000000
 1.0000000000E 00                   0    2.0000000000E 00    2.0000000000
 4.0000000000E 00    3.0000000000E 00                   0    3.000000000
 5.0000000000E 00    4.0000000000E 00    3.0000000000E 00

U-PRIME MATRIX

  1
-3.4999999998E 00 4.2499999997E 00-2.7499999998E 00 2.2499999999E 00
  2
 3.4999999998E 00-7.7500000000E 00 4.2499999999E 00-2.7499999999E 00
  3
-4.9999999964E-01 1.7499999995E 00-2.2499999998E 00 7.4999999986E-01
  4
-1.2914824766E-10 5.0000000010E-01 4.9999999990E-01-1.4999999997E 00

V-PRIME MATRIX

  1
-6.6666666666E-01-3.3333333337E-01-1.6666666667E-01-1.6666666668E-01
  2
-3.7500000001E-01-3.7500000002E-01-2.8124999999E-01-2.1875000001E-01
  3
 4.1666666671E-02-2.9166666669E-01-8.0208333333E-01-3.6458333334E-01
  4
-2.9166666667E-01 4.1666666656E-02-1.3541666668E-01-6.9791666666E-01

VPT X VP

  1
 6.7187500002E-01 3.3854166670E-01 2.2265625001E-01 3.8151041669E-01
  2
 3.3854166670E-01 3.3854166673E-01 3.8932291670E-01 2.1484375004E-01
  3
 2.2265625001E-01 3.8932291670E-01 7.6855468751E-01 4.7623697920E-01
  4
 3.8151041669E-01 2.1484375004E-01 4.7623697920E-01 6.9563802086E-01

VP X VPT

  1
 6.1111111115E-01 4.5833333338E-01 2.6388888892E-01 3.1944444447E-01
  2
 4.5833333338E-01 4.0820312502E-01 3.9908854168E-01 2.8450520835E-01
  3
 2.6388888892E-01 3.9908854168E-01 8.6306423615E-01 3.3875868057E-01
  4
 3.1944444447E-01 2.8450520835E-01 3.3875868057E-01 5.9223090279E-01

UPT X UP

  1
 2.4749999996E 01-4.2874999995E 01 2.5624999997E 01-1.7874999997E 01
  2
-4.2874999995E 01 8.1437499993E 01-4.8312499996E 01 3.1437499997E 01
  3
 2.5624999997E 01-4.8312499996E 01 3.0937499998E 01-2.0312499997E 01
  4
-1.7874999997E 01 3.1437499997E 01-2.0312499997E 01 1.5437499997E 01

UP X UPT

  1
 4.2937499994E 01-6.3062499994E 01 1.7062499994E 01-2.6249999979E 00
  2
-6.3062499994E 01 9.7937499997E 01-2.6937499993E 01 2.3749999970E 00
  3
 1.7062499994E 01-2.6937499993E 01 8.9374999970E 00-1.3749999992E 00
  4
-2.6249999979E 00 2.3749999970E 00-1.3749999992E 00 2.7499999990E 00

VPT X UP

  1
 9.9999999991E-01 2.5829649530E-10-5.8207660913E-11-2.9103830457E-11
  2
-7.4502774320E-11 1.0000000003E 00-1.9099388737E-11 2.3646862246E-11
  3                                                                      = I
-2.2261777606E-10 3.6379788070E-10 9.9999999993E-01 6.5483618528E-11
  4
-2.2642628509E-11 1.7462298274E-10 3.6379788071E-11 9.9999999979E-01
```

Only one test is shown for REVEIG2 since the results present no particular significance except that the matrix K is not skew-symmetric as was anticipated.

```
UP MATRIX,     =  4

   9.40671799 2E-02    -2.3754675249E-01     3.7708598062E-01    -8.9024101642
   3.1644217544E-01    -6.2782339524E-01     5.6007619309E-01     4.3819719532
   5.5556147215E-01    -4.3075473214E-01    -7.0046990430E-01    -1.2305981130
   7.6312984866E-01     6.0320782939E-01     2.3122009924E-01     1.7619122150

G MATRIX,    N=  4

   2.0000000000E 00    4.0000000000E 00    6.0000000000E 00    8.000000000C

UPP MATRIX

   1
 1.8813435985E-01-4.7509350497E-01 7.5417196124E-01-1.7804820329E 00
   2
 1.2657687018E 00-2.5112935810E 00 2.2403047724E 00 1.7527887814E 00
   3
 3.3333688329E 00-2.5845283929E 00-4.2028194258E 00-7.3835886780E-01
   4
 6.1050387893E 00 4.8256626353E 00 1.8497607940E 00 1.4095297720E-01

K MATRIX

   1
 9.7463447412 -01 2.9933249680E-01 8.7835902714E-01-2.2909850124E-02
   2
 2.8746219294E-02 1.0799784214E-01-7.1805015088E-01-2.7720428801E-02
   3
-8.4402690113 -02 8.4764688716E-01-1.9359635358E-01 3.1397766744E-04
   4
 1.5098146321E-01 7.9680847396E-01 2.7215809627E-01 9.9224176882E-01

B MATRIX
 2.4999999999E-01 6.2500000000E-02 2.7777777777E-02 1.5625000000E-02
```